ALAGAPPA UNIVERSITY

(Accredited with 'A+' Grade by NAAC (with CGPA: 3.64) in the Third Cycle and Graded as category - I University by MHRD-UGC)
(A State University Established by the Government of Tamilnadu)

KARAIKUDI – 630 003

DIRECTORATE OF DISTANCE EDUCATION

M.Sc (Computer Science)

Second Year – Fourth Semester

34141– DATA MINING AND WAREHOUSING

Author:
Dr.S.Santhoshkumar
Assistant Professor in Computer Science
Department of Computer Science
Alagappa University,
Karaikudi.630 003.

Reviewer:

SYLLABI-BOOK MAPPING TABLE
DATA MINING AND WAREHOUSING

UNIT	Syllabi	Mapping in Book
	BLOCK 1 **DATA WAREHOUSING**	
1	**Introduction :** Definition – Architecture - warehouse schema warehouse server- OLAP operations	**Pages 11-24**
2	**Data warehouse technology :** Hardware and operating system - warehousing software - Extraction Tools - Transformation tools	**Pages 25-34**
3	**Case Studies:** Data warehousing in Government - tourism - Industry - Genomics data	**Pages 35-45**
	BLOCK 2 **DATA MINING**	
4	**Introduction To Data Mining :** Definition of Data Mining - Techniques in Data Mining - Current trends in data mining	**Pages 46-56**
5	**Different forms of Knowledge :** Data Selection - Data Cleaning - Data Integration - Data Transformation - Data Reduction - Data Enrichment	**Pages 57-63**
6	**Data :**Types of Data –Data Quality – Data Preprocessing - Measures of Similarity and Dissimilarity –Exploration	**Pages 64-96**
	BLOCK 3 **ASSOCIATION RULES**	
7	**Introduction to Association Rule Algorithm:** Methods to Discover Association Rule - A Priori Algorithm - Partition Algorithm - Pincer Search Algorithm	**Pages 97-112**
8	**Dynamic Itemset and FP Tree Growth Algorithm:** Dynamic Item set Algorithm - FP Tree Growth Algorithm	**Pages113-125**
9	**Classification :** Decision Tree Classification - Bayesian Classification - Classification by Back Propagation	**Pages 126-139**
	BLOCK 4 **CLUSTERING TECHNIQUES**	
10	**Introduction To Clustering:** Clustering Paradigms - Partitioning Algorithm- K-Mean Algorithm- K-Medoid Algorithm – CLARA – CLARANS - Hierarchical Clustering -	**Pages140-158**

CONTENTS PAGE NUMBER

BLOCK 2 : DATA MINING

BLOCK – I DATA WAREHOUSING

UNIT 1 INTRODUCTION

Structure

1. 1 Introduction

Data warehousing provides architectures and tools for business executives to systematically organize, understand, and use their data to make strategic decisions. Data warehouse systems are valuable tools in today's competitive, fast-evolving world. In the last several years, many firms have spent millions of dollars in building enterprise-wide data warehouses. Many people feel that with competition mounting in every industry, data warehousing is the latest must-have marketing weapon—a way to retain customers by learning more about their needs. *"Then, what exactly is a data warehouse?"* Data warehouses have been defined in many ways, making it difficult to formulate a rigorous definition. Loosely speaking, a data warehouse refers to a data repository that is maintained separately from an organization's operational databases.

Data warehouse systems allow for integration of a variety of application systems. They support information processing by providing a solid platform of consolidated historic data for analysis.

According to William H. In mon, a leading architect in the construction of data warehouse systems, "A data warehouse is a subject-oriented, integrated, time-variant, and nonvolatile collection of data in support of management's decision making process". This short but comprehensive definition presents the major features of a data warehouse. The four keywords—*subject-oriented, integrated, time-variant,* and *nonvolatile*—distinguish data warehouses from other data repository systems, such as relational database systems, transaction processing systems, and file systems.

1.2 Objectives

After going through the unit you will be able to;

- Understand the fundamentals of Data warehousing
- Know about the warehouse schema
- Discuss about warehouse server
- Learn OLAP operations

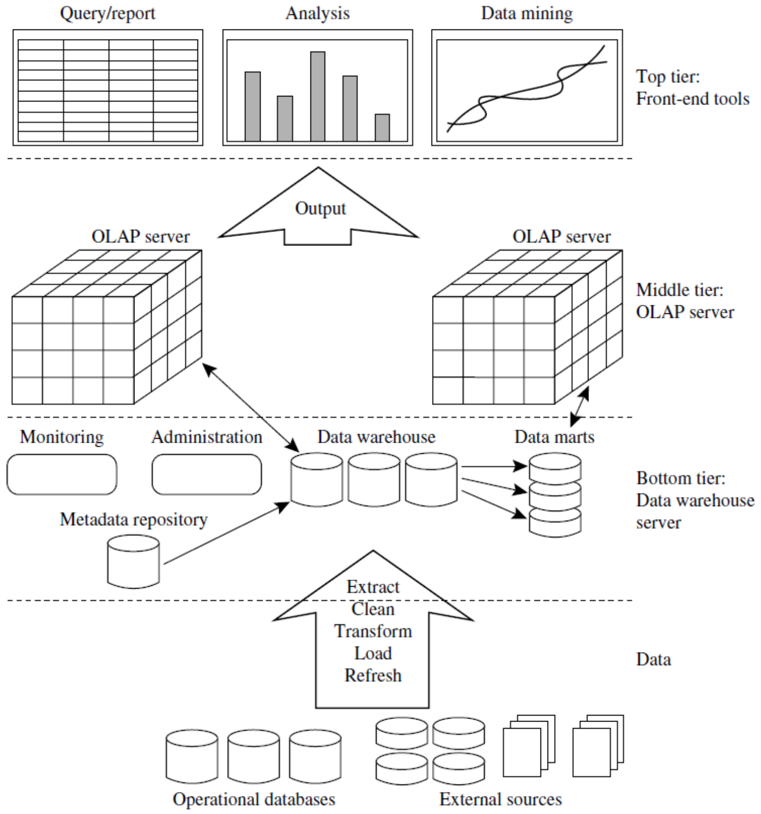
1.3 Definition

A data warehousing is defined as a technique for collecting and managing data from varied sources to provide meaningful business insights. It is a blend of technologies and components which aids the strategic use of data. It is electronic storage of a large amount of information by a business which is designed for query and analysis instead of transaction processing. It is a process of transforming data into information and making it available to users in a timely manner to make a difference.

Data warehouse system is also known by the following name:

- Decision Support System (DSS)
- Executive Information System
- Management Information System
- Business Intelligence Solution
- Analytic Application
- Data Warehouse

1.4 Architecture



The bottom tier is a **warehouse database server** that is almost always a relational database system. Back-end tools and utilities are used to feed data into the bottom tier from operational databases or other external sources (e.g., customer profile information provided by external consultants). These tools and utilities perform data extraction, cleaning, and transformation (e.g., to merge similar data from different sources into a unified format), as well as load and refresh functions to update the data warehouse The data are extracted using application program interfaces known as **gateways**. A gateway is supported by the underlying DBMS and allows client programs to generate SQL code to be executed at a server. Examples of gateways include ODBC (Open Database Connection) and OLEDB (Object Linking and Embedding Database) by Microsoft and JDBC (Java Database Connection). This tier also contains a metadata repository, which stores information about the data warehouse and its contents.

The middle tier is an **OLAP server** that is typically implemented

using either (1) a **relational OLAP(ROLAP)** model (i.e., an extended relational DBMS that maps operations on multidimensional data to standard relational operations); or (2) a **multidimensional OLAP (MOLAP)** model (i.e., a special-purpose server that directly implements multidimensional data and operations). The top tier is a **front-end client layer**, which contains query and reporting tools, analysis tools, and/or data mining tools (e.g., trend analysis, prediction, and so on).

1.5 Warehouse Schema

Warehouse Schema

There are three types of schemas are available in the data warehouse.

- Star Schema
- Snow Flake Schema
- Fact Constellation Schema

Out of which the star schema is mostly used in the data warehouse designs. The second most used data warehouse schema is snow flake schema.

When we consider an example of an organization selling products throughout the world, the main four major dimensions are the product, location, time and organization. Dimension tables have been explained in detail under the section Dimensions. With this example, we will try to provide a detailed explanation about STAR SCHEMA.

What is a Star Schema?

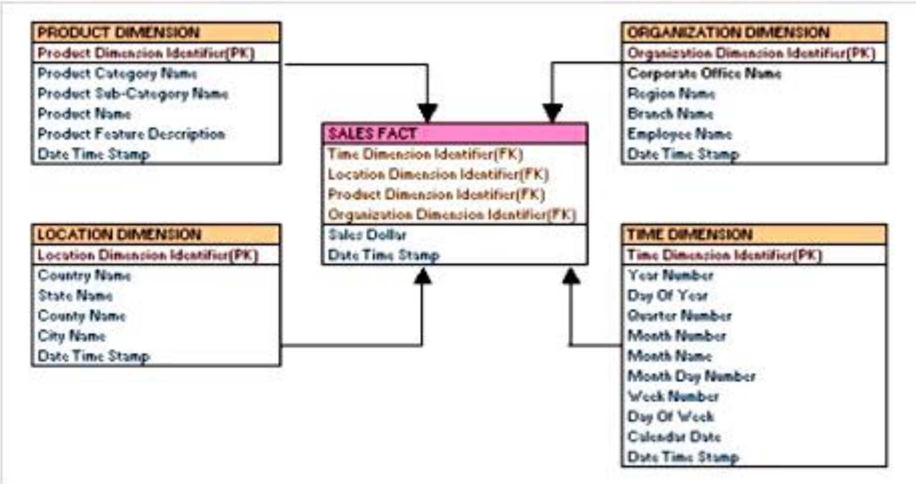
Star Schema is a relational database schema for representing multidimensional data. It is the simplest form of data warehouse schema that contains one or more dimensions and fact tables. It is called a star schema because the entity-relationship diagram between dimensions and fact tables resembles a star where one fact table is connected to multiple dimensions. The center of the star schema consists of a large fact table, and it points towards the dimension tables. The advantage of star schema is slicing down, performance increase and easy understanding of data.

Steps in designing Star Schema

- Identify a business process for analysis (like sales).
- Identify measures or facts (sales dollar).
- Identify dimensions for facts (product dimension, location dimension, time dimension, organization dimension).

- List the columns that describe each dimension.(region name, branch name, region name).
- Determine the lowest level of summary in a fact table (sales dollar).

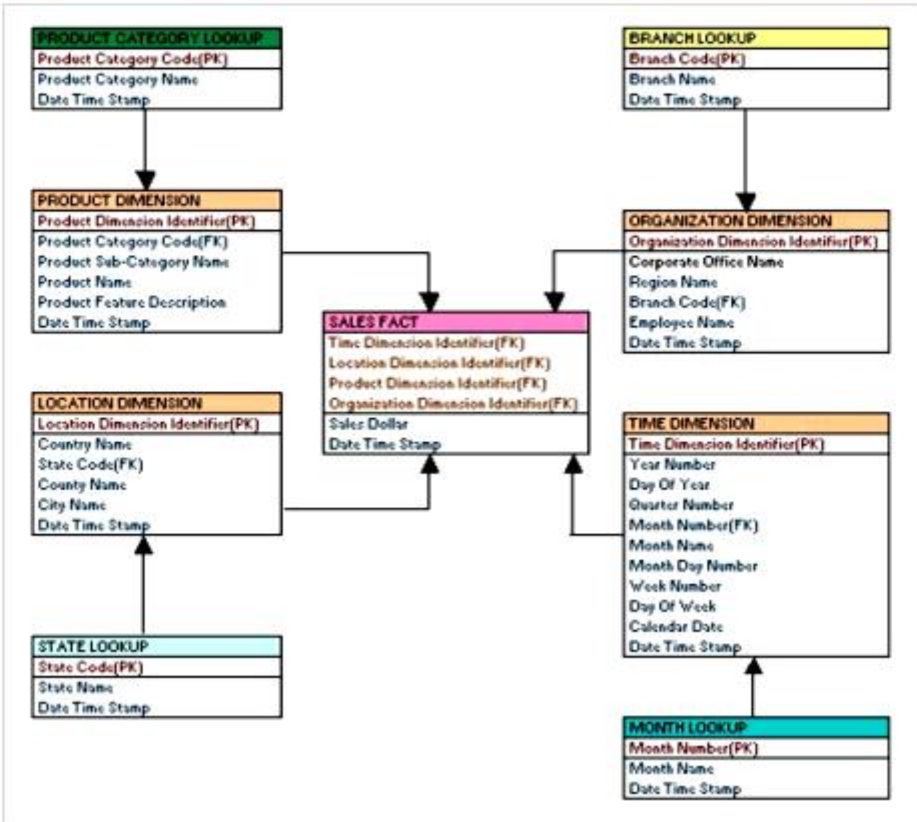
Example of a Star Schema



What is a Snowflake Schema

A snowflake schema is a term that describes a star schema structure normalized through the use of outrigger tables. i.e dimension table hierarchies are broken into simpler tables. In star schema example we had 4 dimensions like location, product, time, organization and a fact table(sales).In Snowflake schema, the example diagram shown below has 4 dimension tables, 4 lookup tables, and 1 fact table. The reason is that hierarchies (category, branch, state, and month) are being broken out of the dimension tables (PRODUCT, ORGANIZATION, LOCATION, and TIME) respectively and shown separately. In OLAP, this Snowflake schema approach increases the number of joins and poor performance in the retrieval of data. In few organizations, they try to normalize the dimension

tables to save space. Since dimension tables hold less space, Snowflake schema approach may be avoided



Fact Table

The centralized table in a star schema is called as the FACT table. A fact table typically has two types of columns: those that contain facts and those that are foreign keys to dimension tables. The primary key of a fact table is usually a composite key that is made up of all of its foreign keys.

In the example fig "Sales Dollar" is a fact(measure) and it can be added across several dimensions. Fact tables store different types of measures like additive, non-additive, and semi-additive measures.

Measure Types

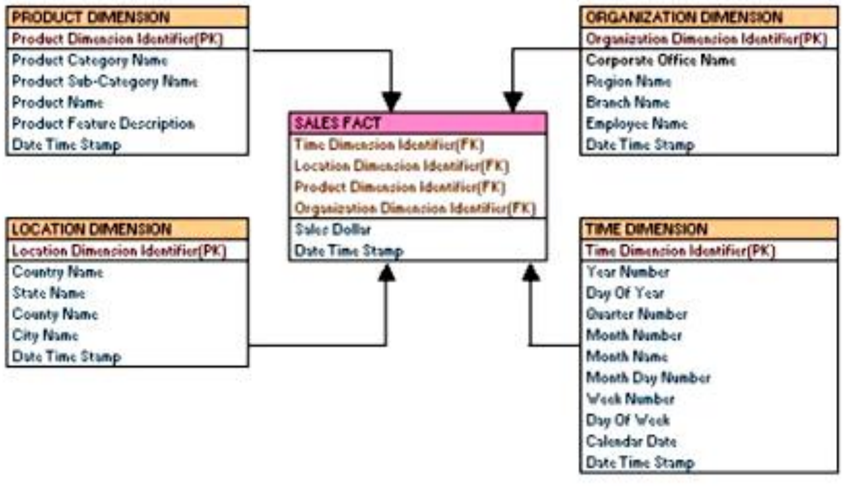
- Additive - Measures that can be added across all dimensions.
- Non-Additive - Measures that cannot be added across all dimensions.
- Semi-Additive - Measures that can be added across few dimensions and not with others.

A fact table might contain either detail level facts or facts that have been aggregated (fact tables that contain aggregated facts are often instead called summary tables).

In the real world, it is possible to have a fact table that contains no measures or facts. These tables are called as **Factless Fact** tables.

Steps in designing Fact Table

- Identify a business process for analysis (like sales).
- Identify measures or facts (sales dollar).
- Identify dimensions for facts (product dimension, location dimension, time dimension, organization dimension).
- List the columns that describe each dimension.(region name, branch name, region name).
- Determine the lowest level of summary in a fact table(sales dollar).



Example of a Fact Table with an Additive Measure in Star Schema:

In the example figure, sales fact table is connected to dimensions location, product, time and organization. Measure "Sales Dollar" in sales fact table can be added across all dimensions independently or in a combined manner which is explained below.

- Sales Dollar value for a particular product
- Sales Dollar value for a product in a location
- Sales Dollar value for a product in a year within a location
- Sales Dollar value for a product in a year within a location sold or serviced by an employee

1.6 Warehouse Server

The warehouse server sits at the core of the architecture described above. We shall discuss different models of the warehouse server. As mentioned earlier, there are three data warehouse models.

Enterprise Warehouse

This model collects all the information about the subjects, spanning the entire organization. It provides corporate-wide data integration, usually from one or more operational systems or external information providers. An enterprise data warehouse requires a traditional mainframe.

Data Marts

Data Marts are partitions of the overall data warehouse. If we visualize the data warehouse as covering every aspect of a company's business (sales, purchasing, payroll, and so forth), then a data mart is a subset of that huge data warehouse built specifically for a department. Data marts may contain some overlapping data. A store sales data mart, for example, would also need some data from inventory and payroll. There are several ways to partition the data, such as by business function or geographic region.

Historically, the implementation of a data warehouse has been limited to the resource constraints and priorities of the MIS organization. The task of implementing a data warehouse can be a very big effort, taking a significant amount of time. And, depending on the implementing alternatives chosen, this could dramatically impact the time it takes to see a payback or return on investment. There are many alternatives to design a data warehouse. One feasible option is to start with a set of data marts for each of the component departments. One can have a stand-alone data mart or a dependent data mart.

The current trend is to define the data warehouse as a conceptual environment. The industry is moving away from a single, physical data warehouse toward a set of smaller, more manageable, databases called data marts. The physical data marts together serve as the conceptual data warehouse. These marts must provide the easiest possible access to information required by its user community.

Stand-alone Data Mart

This approach enables a department or work-group to implement a data mart with minimal or no impact on the enterprise's operational database.

Dependent Data Mart

This approach is similar to the stand-alone data mart, except that management of the data sources by the enterprise database is required. These data sources include operational databases and external sources of data.

remote data sources including major RDBMSs.

The virtual data warehouse scheme lets a client application access data distributed across multiple data sources through a single SQL statement, a single interface. All data sources are accessed as though they are local users and their applications do not even need to know the physical location of the data.

There is a great benefit in starting with a virtual warehouse, since many organizations do not want to replicate information in the physical data warehouse. Some organizations decide to provide both by creating a data warehouse containing summary-level data with access to legacy data for transaction details.

A virtual database is easy and fast, but it is not without problems. Since the queries must compete with the production data transactions, its performance can be considerably degraded. Since there is no metadata, no summary data or history, all the queries must be repeated, creating an additional burden on the system. Above all, there is no clearing or refreshing process involved, causing the queries to become very complex.

1.7 OLAP Operations

Typical OLAP Operations

"How are concept hierarchies useful in OLAP?" In the multidimensional model, data are organized into multiple dimensions, and each dimension contains multiple levels of abstraction defined by concept hierarchies. This organization provides users with the flexibility to view data from different perspectives. A number of OLAP data cube operations exist to materialize these different views, allowing interactive querying and analysis of the data at hand. Hence, OLAP provides a user-friendly environment for interactive data analysis.

OLAP operations.

Let's look at some typical OLAP operations for multidimensional data. Each of the following operations described is illustrated in Figure 4.12. At the center of the figure is a data cube for *All Electronics* sales. The cube contains the dimensions *location, time,* and *item*, where *location* is aggregated with respect to city values, *time* is aggregated with respect to quarters, and *item* is aggregated with respect to item types.

To aid in our explanation, we refer to this cube as the central cube. The measure displayed is *dollars sold* (in thousands). (For improved readability, only some of the cubes' cell values are shown.) The data examined are for the cities Chicago, New York, Toronto, and Vancouver.

Roll-up:

The roll-up operation (also called the *drill-up* operation by some vendors) performs aggregation on a data cube, either by *climbing*

up a concept hierarchy for a dimension or by *dimension reduction*. Figure 4.12 shows the result of a roll-up operation performed on the central cube by climbing up the concept hierarchy for *location* given in Figure. This hierarchy was defined as the total order "*street* < *city* < *province or state* < *country*." The roll-up operation shown aggregates the data by ascending the *location* hierarchy from the level of *city* to the level of *country*. In other words, rather than grouping the data by city, the resulting cube groups the data by country.

When roll-up is performed by dimension reduction, one or more dimensions are removed from the given cube. For example, consider a sales data cube containing only the *location* and *time* dimensions. Roll-up may be performed by removing, say, the *time* dimension, resulting in an aggregation of the total sales by location, rather than by location and by time.

Drill-down:

Drill-down is the reverse of roll-up. It navigates from less detailed data to more detailed data. Drill-down can be realized by either *stepping down a concept hierarchy* for a dimension or *introducing additional dimensions*. Figure 4.12 shows the result of a drill-down operation performed on the central cube by stepping down a concept hierarchy for *time* defined as "*day* < *month* < *quarter* < *year*." Drill-down occurs by descending the *time* hierarchy from the level of *quarter* to the more detailed level of *month*. The resulting data cube details the total sales per month rather than summarizing them by quarter. Because a drill-down adds more detail to the given data, it can also be performed by adding new dimensions to a cube. For example, a drill-down on the central cube of Figure 4.12 can occur by introducing an additional dimension, such as *customer group*.

Slice and dice:

The *slice* operation performs a selection on one dimension of the given cube, resulting in a sub cube. Figure 4.12 shows a slice operation where the sales data are selected from the central cube for the dimension *time* using the criterion *time* D "Q1." The *dice* operation defines a sub cube by performing a selection on two or more dimensions. Figure 4.12 shows a dice operation on the central cube based on the following selection criteria that involve three dimensions: (*location* D "Toronto" or "Vancouver") and (*time* D "Q1" or "Q2") and (item D "home entertainment" or "computer").

Pivot (rotate):

Pivot (also called *rotate*) is a visualization operation that rotates the Data axes in view to provide an alternative data presentation. Figure 4.12 shows a pivot operation where the *item* and *location* axes in a 2-D slice are rotated. Other examples include rotating the axes in 3-D cube, or transforming a 3-D cube into a series of 2-D planes.

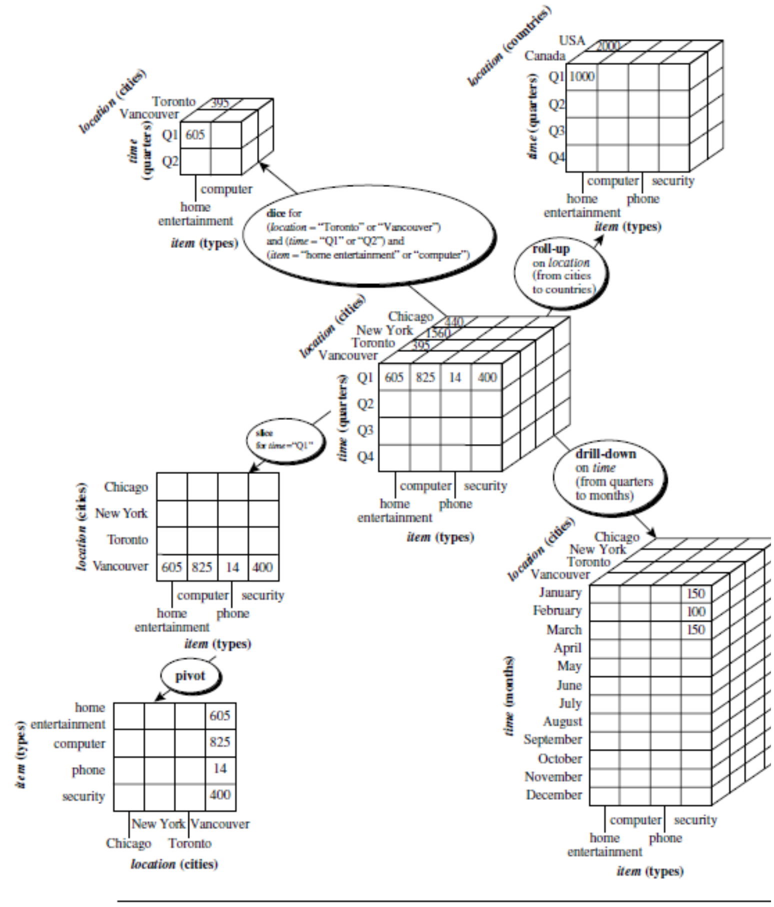


Figure 4.12 Examples of typical OLAP operations on multidimensional data.

Other OLAP operations:

Some OLAP systems offer additional drilling operations. For example, **drill-across** executes queries involving (i.e., across) more than one fact table. The **drill-through** operation uses relational SQL facilities to drill through the bottom level of a data cube down to its back-end relational tables. Other OLAP operations may include ranking the top *N* or bottom *N* items in lists, as well as computing moving averages, growth rates, interests, internal return rates, depreciation, currency conversions, and statistical functions.

OLAP offers analytical modeling capabilities, including a calculation engine for deriving ratios, variance, and so on, and for computing measures across multiple dimensions. It can generate summarizations, aggregations, and hierarchies at each granularity level and at every dimension intersection. OLAP also supports functional models for forecasting, trend analysis, and statistical analysis. In this context, an OLAP engine is a powerful data anal

1.8 Check Your Progress

1. What is Data warehousing?
2. List Functions of Data warehouse Tools and Utilities
3. What is Data Cube?

1.9 Answers to Check Your Progress Questions

1. **Data warehouse**

A data warehousing is defined as a technique for collecting and managing data from varied sources to provide meaningful business insights.

2. **Functions of Data Warehouse Tools and Utilities**

- Data Extraction − Involves gathering data from multiple heterogeneous sources.
- Data Cleaning − Involves finding and correcting the errors in data.
- Data Transformation − Involves converting the data from legacy format to warehouse format.
- Data Loading − Involves sorting, summarizing, consolidating, checking integrity, and building indices and partitions.
- Refreshing − Involves updating from data sources to warehouse.

3. **Data Cube**

A data cube helps us represent data in multiple dimensions. It is defined by dimensions and facts. The dimensions are the entities with respect to which an enterprise preserves the records.

1.8 Summary

A data warehouses provides us generalized and consolidated data in multidimensional view. Along with generalized and consolidated view of data, a data warehouses also provides us Online Analytical Processing (OLAP) tools. These tools help us in interactive and effective analysis of data in a multidimensional space. This analysis results in data generalization and data mining.

1.9 Key Words

Data Warehouse: It is a federated repository for all the **data** collected by an enterprise's various operational systems.

Schema:

Schema is a logical description of the entire database. It includes

the name and description of records of all record types including all associated data-items and aggregates

OLAP:
(Online Analytical Processing) is the technology behind many Business Intelligence (BI) applications. OLAP is a powerful technology for data discovery, including capabilities for limitless report viewing, complex analytical calculations, and predictive “what if” scenario (budget, forecast) planning

1.10 Self-Assessment Questions and Exercises

1. Describe the Architecture of Data Warehouse?
2. Explain in detail about OLAP Operations?
3. Explain in detail about Warehouse schema.

1.11 Further Readings

1. Pang-Ning Tan, Michael Steinbach and Vipin Kumar, "Introduction to Data Mining", Person Education, 2007.
2. K.P. Soman, Shyam Diwakar and V. Aja, "Insight into Data Mining Theory and Practice", Eastern Economy Edition, Prentice Hall of India, 2006.
3. G. K. Gupta, "Introduction to Data Mining with Case Studies", Eastern Economy Edition, Prentice Hall of India, 2006.
4. Daniel T.Larose, "Data Mining Methods and Models", Wiley-Interscience, 2006.
5. Alex Berson and Stephen J.Smith, "Data Warehousing, Data Mining and OLAP", Tata McGraw – Hill Edition, Thirteenth Reprint 2008.
6. Jiawei Han and Micheline Kamber, "Data Mining Concepts and Techniques", Third Edition, Elsevier, 2012.
7. Pieter Adriaans, Dolf Zantinge Data Mining, Pearson Education
8. George M. Marakas Modern Data Warehousing, Mining, and Visualization: Core Concepts, Prentice Hall, 1st edition
9. Margaret H. Dunham Data Mining, Prentice Hall, 1st edition,
10. David J. Hand Principles of Data Mining (Adaptive Computation and Machine Learning), Prentice Hall, 1st edition
11. Michael J. Corey, Michael Abbey, Ben Taub, Ian Abramson Oracle 8i Data Warehousing McGraw-Hill Osborne Media, 2nd edition

UNIT – 2
DATA WAREHOUSE TECHNOLOGY

Structure

2.1 Introduction

In the 1990s, as business grew more complex, corporate offices spread across the globe, and competition became fiercer, business executives became desperate for information to be competitive and improve the bottom line. The operational computer systems did provide information to run day-to-day operations, but what the executives needed were different kinds of information that could be readily used to make strategic decisions. They wanted to know where to build the next warehouse for their product and which markets they should strengthen. The operational systems, important as they were, could not provide strategic information. Due to rapidly changed market dynamics, competitive pressure, globalization and other similar factors forced business to review their structures, approaches and strategies. Therefore, businesses were compelled to look into new ways of getting information for dynamic markets. During the last decade, the interest to analyze data has increased significantly, because of competitive advantages of data in decision making process. A key to survival in the business world is being able to analyze, plan and react to changing business conditions as fast as possible. Many organizations own billions of bytes of data, but they suffer from different problems because data are spread over different computer systems, data from

different sources are incompatible, data are available too late, etc. In order to solve these problems, the new concepts and tools have evolved into an information technology called Data Warehousing. The Data Warehouse (DW) can meet informational needs of knowledge workers and can provide strategic business opportunities by allowing customers and vendors to access corporate data. A large retail store collects vast amounts of information about their day to day activities. The same retail store probably collects other types of information as well, such as customer data, inventory data, advertisement data, employee data, etc. An increasing number of organizations are realizing that the vast amounts of collected data can and must be used to guide their business decisions. Typically, the management of the organization wants to answer complex analytical queries based on the collected data.
Building a Data Warehouse provides a number of benefits such as:

• The processing of analytical queries is simplified because only the data warehouse needs to be accessed.

• The warehouse data can keep a historical record of the various source data. By retaining all of this data, the current activity of an organization can be compared against history and can also be used for forecasting the future activities of an organisation.

2.2 Objectives

After going through the unit you will be able to;

- Understand about the Hardware and OS
- Know about Warehousing software
- Explain about Extraction Tools
- List the names of Transformations Tools

2.3 Hardware and Operating System

Hardware and operating systems make up the computing environment for your data warehouse. All the data extraction, transformation, integration, and staging jobs run on the selected hardware under the chosen operating system. When you transport the consolidated and integrated data from the staging area to your data warehouse repository, you make use of the server hardware and the operating system software. When the queries are initiated from the client workstations, the server hardware, in conjunction with the database software, executes the queries and produces the results. Here are some general guidelines for hardware selection, not entirely specific to hardware for the data warehouse.

Scalability.
When your data warehouse grows in terms of the number of users, the number of queries, and the complexity of the queries, ensure that your selected hardware could be scaled up.

Support.
Vendor support is crucial for hardware maintenance. Make sure that the support from the hardware vendor is at the highest possible level.

Vendor Reference. It is important to check vendor references with other sites using hardware from this vendor. You do not want to be caught with your data warehouse being down because of hardware malfunctions when the CEO wants some critical analysis to be completed.

Vendor Stability. Check on the stability and staying power of the vendor. The term *hardware and operating systems* refers to the server platforms and operating systems that serve as the computing environment of the data warehouse. Warehousing environments are typically separate from the operational computing environments (i.e., a different machine is used) to avoid potential resource contentions between operational and decisional processing. Enterprises are correctly wary of computing solutions that may compromise the performance levels of mission-critical operational systems.

2.4 Warehousing Software

Data warehousing software runs the databases that make up a company's **data warehouse**. As the single source of historical truth for the combined **data** of many different **software** tools from across the company, the **data warehouse** makes up the central **data** store for running business intelligence **software**.

A data warehouse is a database designed for data analysis instead of standard transactional processing. A data warehouse acts as a conduit between operational data stores and supports analytics on the composite data. Slices of data from the warehouse—e.g. summary data for a single department to use, like sales or finance—are stored in a “data mart” for quick access.

In order for a data warehouse to support decision-making effectively, data extracted from various data sources and loaded into the warehouse is normalized. It can be organized into tables, cleaned of redundancy and transformed for consistency. The process by which this happens is called Extract, Transform, and Load (ETL). Once appropriately structured data is made available for querying and analysis.

Data Warehouse Features & Capabilities

To support analyses data warehouses provide the following capabilities:

- Associated input, extract, and data management tools for preparation
- Extract from a multitude of source file types (flat files, excel, application data, etc.)
- May load & normalize structured, semi-structured, or unstructured data
- Data transformation (cleansing, deduplication, consistency)
- Data reconciliation for various naming conventions
- Native & autonomous storage and processing optimization
- Provide a 360 view of all enterprise data
- Multiple deployment options (private or public cloud, on-premise, hybrid cloud)
- Available as-a-service (automated infrastructure management)
- Integrated machine-learning algorithms, AI
- Access controlled data sharing, data mart
- Deploy virtualized data warehouse for extra security, access control
- In-built data encryption for high-security needs

2.5 Extraction Tools

Extraction is the operation of extracting data from a source system for further use in a data warehouse environment. This is the first step of the ETL process. After the extraction, this data can be transformed and loaded into the data warehouse.

The source systems for a data warehouse are typically transaction processing applications. For example, one of the source systems for a sales analysis data warehouse might be an order entry system that records all of the current order activities. Designing and creating the extraction process is often one of the most time-consuming tasks in the ETL process and, indeed, in the entire data

warehousing process. The source systems might be very complex and poorly documented, and thus determining which data needs to be extracted can be difficult. The data has to be extracted normally not only once, but several times in a periodic manner to supply all changed data to the data warehouse and keep it up-to-date. Moreover, the source system typically cannot be modified, nor can its performance or availability be adjusted, to accommodate the needs of the data warehouse extraction process.

These are important considerations for extraction and ETL in general. This chapter, however, focuses on the technical considerations of having different kinds of sources and extraction methods. It assumes that the data warehouse team has already identified the data that will be extracted, and discusses common techniques used for extracting data from source databases.

The extraction method you should choose is highly dependent on the source system and also from the business needs in the target data warehouse environment. Very often, there is no possibility to add additional logic to the source systems to enhance an incremental extraction of data due to the performance or the increased workload of these systems. Sometimes even the customer is not allowed to add anything to an out-of-the-box application system.

Logical Extraction Methods

There are two types of logical extraction:

- Full Extraction
- Incremental Extraction

Full Extraction

The data is extracted completely from the source system. Because this extraction reflects all the data currently available on the source system, there's no need to keep track of changes to the data source since the last successful extraction. The source data will be provided as-is and no additional logical information (for example, timestamps) is necessary on the source site. An example for a full

extraction may be an export file of a distinct table or a remote SQL statement scanning the complete source table.

Incremental Extraction

At a specific point in time, only the data that has changed since a well-defined event back in history will be extracted. This event may be the last time of extraction or a more complex business event like that last booking day of a fiscal period. To identify this delta change there must be a possibility to identify all the changed information since this specific time event. This information can be either provided by the source data itself such as an application column, reflecting the last-changed timestamp or a change table where an appropriate additional mechanism keeps track of the changes besides the originating transactions. In most cases, using the latter method means adding extraction logic to the source system.

Many data warehouses do not use any change-capture techniques as part of the extraction process. Instead, entire tables from the source systems are extracted to the data warehouse or staging area, and these tables are compared with a previous extract from the source system to identify the changed data. This approach may not have significant impact on the source systems, but it clearly can place a considerable burden on the data warehouse processes, particularly if the data volumes are large.

Oracle's Change Data Capture (CDC) mechanism can extract and maintain such delta information. "Change Data Capture" for further details about the Change Data Capture framework.

Physical Extraction Methods

Depending on the chosen logical extraction method and the capabilities and restrictions on the source side, the extracted data can be physically extracted by two mechanisms. The data can either be extracted online from the source system or from an offline structure. Such an offline structure might already exist or it might be generated by an extraction routine.

There are the following methods of physical extraction:

- Online Extraction
- Offline Extraction

Online Extraction

The data is extracted directly from the source system itself. The extraction process can connect directly to the source system to access the source tables themselves or to an intermediate system that stores the data in a preconfigured manner (for example, snapshot logs or change tables). Note that the intermediate system is not necessarily physically different from the source system. With online extractions, you need to consider whether the distributed transactions are using original source objects or prepared source objects.

Offline Extraction

The data is not extracted directly from the source system but is staged explicitly outside the original source system. The data already has an existing structure (for example, redo logs, archive logs or transportable tablespaces) or was created by an extraction routine.

You should consider the following structures:

- Flat files Data in a defined, generic format. Additional information about the source object is necessary for further processing.
- Dump files Oracle-specific format. Information about the containing objects may or may not be included, depending on the chosen utility.
- Redo and archive logs Information is in a special, additional dump file.

A powerful way to extract and move large volumes of data between Oracle databases. Oracle recommends that you use transportable table spaces whenever possible, because they can provide considerable advantages in performance and manageability over other extraction techniques.

2.6 Transformations Tools

Data transformations are often the most complex and, in terms of processing time, the most costly part of the extraction, transformation, and loading (ETL) process. They can range from simple data conversions to extremely complex data scrubbing techniques. Many, if not all, data transformations can occur within

an Oracle database, although transformations are often implemented outside of the database (for example, on flat files) as well. This chapter introduces techniques for implementing scalable and efficient data transformations within the Oracle Database. The examples in this chapter are relatively simple. Real-world data transformations are often considerably more complex. However, the transformation techniques introduced in this chapter meet the majority of real-world data transformation requirements, often with more scalability and less programming than alternative approaches. This chapter does not seek to illustrate all of the typical transformations that would be encountered in a data warehouse, but to demonstrate the types of fundamental technology that can be applied to implement these transformations and to provide guidance in how to choose the best techniques.

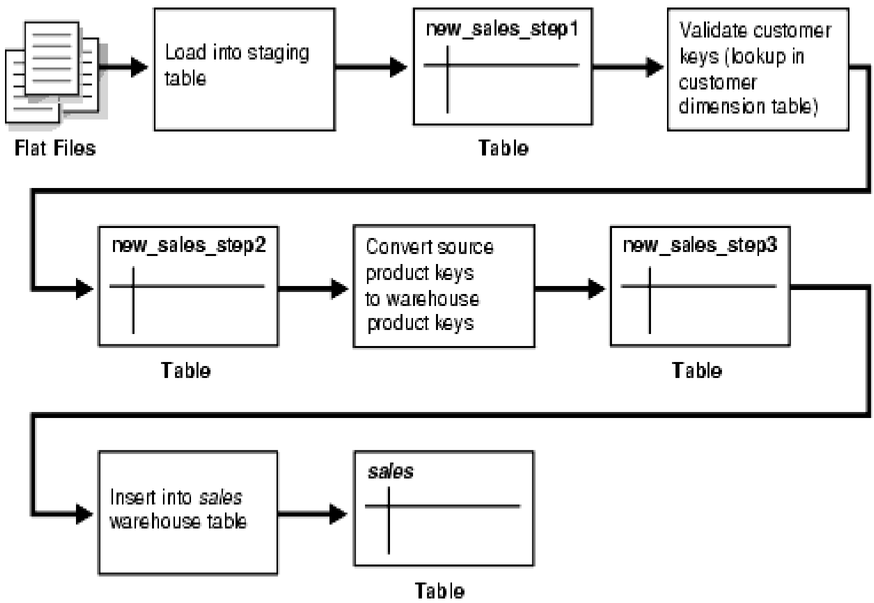
Transformation Flow

From an architectural perspective, you can transform your data in two ways:

- Multistage Data Transformation
- Pipelined Data Transformation

Multistage Data Transformation

The data transformation logic for most data warehouses consists of multiple steps. For example, in transforming new records to be inserted into a sales table, there may be separate logical transformation steps to validate each dimension key.



Description of "Figure 1 Multistage Data Transformation"

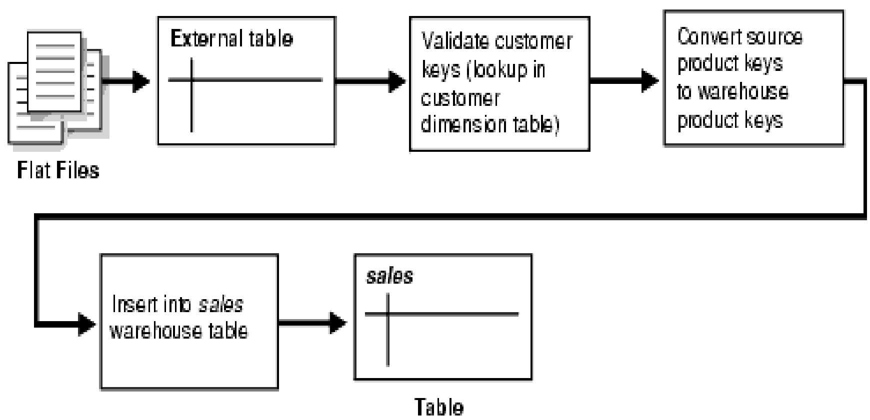
When using Oracle Database as a transformation engine, a common strategy is to implement each transformation as a separate SQL operation and to create a separate, temporary staging table (such as the tables new_sales_step1 and new_sales_step2 in Figure 1) to store the incremental results for each step. This load-then-transform strategy also provides a natural check pointing scheme to the entire transformation process, which enables to the process to be more easily monitored and restarted. However, a disadvantage to multi staging is that the space and time requirements increase.

It may also be possible to combine many simple logical transformations into a single SQL statement or single PL/SQL procedure. Doing so may provide better performance than performing each step independently, but it may also introduce difficulties in modifying, adding, or dropping individual transformations, as well as recovering from failed transformations.

Pipelined Data Transformation

The ETL process flow can be changed dramatically and the database becomes an integral part of the ETL solution. The new functionality renders some of the former necessary process steps obsolete while some others can be remodeled to enhance the data flow and the data transformation to become more scalable and non - interruptive. The task shifts from serial transform-then-load process (with most of the tasks done outside the database) or load- then-transform process, to an enhanced transform-while-loading.
Oracle offers a wide variety of new capabilities to address all the issues and tasks relevant in an ETL scenario. It is important to understand that the database offers toolkit functionality rather than trying to address a one-size-fits-all solution. The underlying database has to enable the most appropriate ETL process flow for a specific customer need, and not dictate or constrain it from a technical perspective. Figure -2 illustrates the new functionality, which is discussed throughout later sections.

Figure -2 Pipelined Data Transformation



2.7 Check your Progress

1. What kind of Extraction Methods Available?
2. List any two Transformation Tools

2.8 Answers to Check Your Progress Questions

1. (i) Logical Extraction Methods
   (ii) Physical Extraction Methods

2. (i) Tera Data (ii) Oracle 12 C

2.9 Summary

- Maintaining past and present records.
- Helping organizations to take effective business decisions with precise data analysis.
- It provides the multidimensional view of consolidated data in a warehouse.
- Additionally, the data warehouse environment supports **ETL (Extraction, Transform and Load)** solutions, data mining capabilities, statistical analysis, reporting and **Online Analytical Processing (OLAP)** Tools, which help in interactive and efficient data analysis in a multifaceted view.

2.10 Key Words

MDT : Multistage Data Transformation
PDT : Pipelined Data Transformation
CDC : Change Data Capture

2.11 Self-Assessment Questions and Exercises

1. What do you meant by MDT?
2. What are two types of Logical Extraction Method?
3. Write short note on : Data warehousing Operating System
4. List any two names of Transformation Tools

2.12 Further Readings

1. Pang-Ning Tan, Michael Steinbach and Vipin Kumar, “Introduction to Data Mining”, Person Education, 2007.
2. K.P. Soman, Shyam Diwakar and V. Aja, “Insight into Data Mining Theory and Practice”, Eastern Economy Edition, Prentice Hall of India, 2006.
3. G. K. Gupta, “Introduction to Data Mining with Case Studies”, Eastern Economy Edition, Prentice Hall of India, 2006.
4. Daniel T.Larose, “Data Mining Methods and Models”, Wiley-Interscience, 2006.
5. Alex Berson and Stephen J.Smith, “Data Warehousing, Data Mining and OLAP”, Tata McGraw – Hill Edition, Thirteenth Reprint 2008.
6. Jiawei Han and Micheline Kamber, “Data Mining Concepts and Techniques”, Third Edition, Elsevier, 2012.

UNIT – 3
CASE STUDIES

Structure

3.1 Introduction

A data warehouse (DW) is a collection of technologies aimed to enabling the decision maker to make better and faster decisions. It is designed to support On Line Analytical Processing

(OLAP). There are some previous works that are related to the general architecture of a data warehouse and the development of data warehouse prototypes in fields like telecommunication, banking, and insurance. Warehousing for the tourism industry. First we focus on the three dimensions of tourism: economic, social and cultural and we point out the major implication of tourism industry in economy. Then we make a short analyze of the information systems actually used in this area and we show that advanced technology used in the different components of this sector is still low. Here is an urgent need for data warehousing in government circles. As the volumes of data grow large and the need for new and innovative information becomes manifest, it becomes apparent that the organization or agency needs a data warehouse. But surprisingly, data warehouses have been slow to be adopted in the government circles. There are some fundamental reasons.

The most basic reason is that there is a significant difference in motivation for data warehousing in the commercial world and the governmental world. In the commercial world, the most fundamental motivations for data warehousing are to increase profit or increase market share protection. There are many other motivations for data warehousing in the commercial world, but these two motivations are the most basic and most visceral.

Government agencies, on the other hand, try to optimize their resources while building a data warehouse to the benefit of the constituency they reach. They are not concerned with reducing the size of their department due to budgetary reasons and political power.

3.2 Objectives

After going through the unit you will be able to;

- Understand about case studies method
- Know about warehousing in Government
- Explain about warehousing in Tourism
- Learn about warehousing in Industry

3.3 Data warehousing in Government

Government agencies, on the other hand, try to optimize their resources while building a data warehouse to the benefit of the constituency they reach. They are not concerned with reducing the size of their department due to budgetary reasons and political power. What are the visceral government motivations

for a government data warehouse? The government motivations for a data warehouse are:

The government motivations for a data warehouse are:

- The need for accuracy of data;
- The need for data at the lowest cost;
- The need for data at the fastest speed; and
- The need for integration of data.

Data warehousing addresses all of these needs. But, there are a lot of forces working against a data warehouse in government circles. Working against the building of the data warehouse in government circles are attitudes such as:

- "Well that's not how we did it 10 years ago." And sure enough,10 years ago you had significantly less data and significantly fewer demands for information on the government organization to produce information.

- "If I bring in a data warehouse, I am not going to need as many people." That's right, you can do things a lot more efficiently and inexpensively when you have a data warehouse.

- "I have to protect my data. No one else can look at it." That's just what the terrorists were planning on.

- "My SI vendor doesn't understand the data warehouse; therefore there must be other ways to get the information."All true. Get a new SI that does understand the data warehouse and information economics. (In fact, when was the last time you got a new SI? Is that healthy?).

- "My tour of duty is only two years. We won't have much of a data warehouse built in that time, so it is going to hurt my chances of promotion to the next rank." This is true for political appointees as well as military personnel.

- "A data warehouse costs so much money. Can't I spend money this year to create some reports?" And next year spend some more money. And the year after that, even more, etc.

  "I only have a limited scope of assignment. A data warehouse goes well beyond my scope." After this it becomes someone else's problem.

And indeed there are hundreds of other reasons in government for not building a proper architecture. All of the excuses have a seed of truth. And all of them pale in comparison to the long-term need for efficient, accurate and cheap information in the government. However, the basic truth and bottom line responsibility for the government remains that it is the steward of the public trust and must organize and maintain the information it accumulates in an efficient, and accessible and meaningful architecture for the future.

3.4 Tourism

Information technology was initially viewed by the tourism industry as a back -office function that supports the finance and accounting areas. The industry has advanced far beyond this view during the past decade. In some sessions tourism industry leaders pondered the role of technology. Among the conclusions reached were: "Going forward, technology will be the most competitive weapon for any touristic company. If touristic organizations want to compete successfully, they must do so by using technology to drive value to both the customer and to the firm

Front Office Information Systems:

Front-office information systems are those data processing systems that provide reports in visual or written form. They are used mainly in the management of tourist accommodation (hotels, motels, hostels or cruise ships) or in the travel agencies activities. These systems may be used for: - tourists registration when the personal data about tourists are collected; - marketing of various tourism products, such as rental cars; - rooms management, when are collected and processed data regarding the rooms status, (allows instant viewing of room availability for all room types, indicates whether rooms are dirty or clean, allows rooms to be placed out of inventory or out of order to restrict rental) - tracks of revenues, providing transaction processing and obtain information about any debts and credits in relation to customers.

Information Systems Used for Reservations:

This kind of systems provides rapid access to information and ensures the accuracy of this information. They bring information services, booking and selling and are used both by individual tourists and travel agents or commissioners. Such systems can be classified into the following categories: - information systems that function as data banks accessible through transmission systems for consultation; - availability systems that provide information on the status of free or completely occupied a location at a time; - computerized reservation systems. Most often this type of systems uses Web technologies. These systems use hardware and software specific to conduct them activities. Although providers of tourist services in Romania currently use such systems for ticketing most, is well to remember that these systems can be used for marketing or management activities. Despite the fact that tourism is a dynamic industry with important implications on the economy to adopt advanced technology of the different components of this sector is still low. For example, according to a study conducted by the magazine e -Business Watch, the percentage of tourism organizations adopting and using application in different areas (such as customer relationship management CRM, enterprise resource planning ERP or supply chain management SCM) is sensibly lower than in other economic and industrial sectors

Information Systems Using Data mining Techniques

In the tourism industry knowing the guests - where they are from, how much they spend, and when and on what they spend it- can help a company to formulate marketing strategies and maximize profits. Due to technological development touristic companies have accumulated large amounts of customer data, which can be organized and integrated in databases that can be used to guide marketing decision. Since identification of important variables and relationships located in these consumer - information systems can be a difficult task, some companies have attempted to raise the power of information by using data mining technologies that exploits the data regarding the consumer. Such data-mining technology allows these companies to predict consumer -behavior trends, which are potentially useful for marketing applications.

3.5 Industry

Client Information

Industry: Financial Services
Size: >10000 employees; >$1 billion in revenue
Areas of Engagement: Information Management, Business Intelligence, Financial Performance Management

The Challenge

The client's IT group was struggling to keep their governance and planning efforts moving at the pace of business. They had taken steps to centralize the many data sources they needed to deal with and get the ball rolling toward accurate, actionable reporting on those assets, but their infrastructure was becoming outdated and was no longer capable of managing the increased information demands the team was seeing or integrating effectively with their business intelligence (BI) reporting processes. They engaged with Iron side to move their existing data warehouse to a more modern solution that would address three key department goals:

- Move warehousing infrastructure out of the current obsolete environment.
- Store historical point-in-time data for more accurate referencing of past events.
- Address the pain points occurring between the data warehouse and the Cognos BI reporting layer, increasing
- query efficiency and enabling more timely analytics.

The Journey

The Ironside Information Management team signed on to work hand in hand with the client's database engineers to transition to a modern data warehouse. Through discovery conversations with IT leadership, the Iron side resources assigned to the project outlined a full-scale migration and redesign plan that would bring approximately 20 tables from an array of data sources, including TM1, A-Track, Remedy, Oracle GL, and Excel, into an IBM Pure Data for Analytics (Netezza) implementation capable of meeting and exceeding the client's requirements. As part of the migration, Iron side was also assigned to reengineer both the ETL processes used to move and transform all the different information streams and the reporting layer making that information available for analysis.

This project is still underway, and Iron side has completed several phases so far:

- Collected requirements and use cases for the new data warehouse.
- Documented the existing warehouse logic, including all data extracts, transformations, schedules, etc.
- Served as the solution architect for a proof of concept of the data warehouse redesign
- Worked with the client's database engineers to set specifications for new ETL workflows to feed the POC design.
- Rebuilt the reporting layer to work seamlessly with the new environment.
- Tested data handling and report output using the POC system.

The Results

The initial findings coming out of the concepting and testing phases of the project are very promising, and Ironside's team is confident that the final solution will deliver the modern data handling functionality that the client needs to continue their success. These encouraging results include:

- Improved ease of use and performance in Cognos reporting.
- Enablement of time comparison reporting, such as evaluating what the data looked like last month versus the present day.
- A performance increase approximately 60 times faster than what was possible with the previous environment was observed during testing.
- Report outputs that used to take around 2 minutes to run are now taking around 2 seconds.

Using this compelling proof of concept as a springboard, Iron side and the client are now making adjustments and preparing to roll out the full-scale data warehouse solution. With this level of data performance at their disposal, the IT team will easily be able to meet the stringent demands of the financial services industry and deliver the kinds of answers that will drive the business forward.

3.6 Genomics Data

BioMart (v0.8 rc6)

Unlike other systems presented in this benchmark, BioMart is a data federation framework that provides a unified user interface to multiple data sources that may be distributed worldwide. One of the main benefits of BioMart is its ability to integrate any existing data source, which is internally represented using their 'reverse star' schema that is optimized for fast retrieval of large amounts of data. The software package also includes Mart Configurator, a user-friendly tool that facilitates the configuration of the web user interface and the definition of the relationships between data sources. It also provides REST/SOAP and Java APIs, as well as SPARQL for semantic queries. BioMart has been successfully used by numerous laboratories and consortia to build integrated portals for cancer-related microarray and gene expression data.

BioXRT (v1.03)

BioXRT was designed to allow biologists to publish their data on the Internet with only minimal knowledge of databases and web development. In particular, BioXRT is reputed an excellent choice for small and medium size laboratories, which need to publish their results and correlate them to data from other public sources. The system was implemented in Perl and allows researchers to convert spreadsheets to the internal XRT data schema into the underlying MySQL database. The XRT schema comprises four Cross-Referenced Tables, which describe data, their structure and their relationships.

The Cross-Referenced Tables (XRT) model is highly flexible and may be expanded as needed to accommodate unforeseen data complexities. BioXRT has been used in various projects, ranging from the annotation of the Human Chromosome to the study of structural variation of chromosomes in autism spectrum disorder, thereby demonstrating the versatility and flexibility of the framework.

InterMine (v1.0b)

InterMine is an open-source framework that features a user-friendly web interface. InterMine relies on a traditional ETL (Extract, Transform and Load) architecture and provides a core data model and a collection of parsers to load

data from 28 typical data sources such as the Gene Ontology the Kyoto Encyclopedia of Genes and Genomes or the Protein Data Bank The core model and the set of parsers may be extended by end users to accommodate new types of data or define new relationships between data. The default user interface can be customized and enhanced with widgets and plugins such as the genome browser GBrowse the interaction graph viewer Cytoscape and gene expression heat maps. The core of InterMine, implemented in Java, translates the data model into a normalized database schema and loads the data into the underlying PostgreSQL database with optional pre/post-processing steps. InterMine also features Java, Perl, Python, Ruby and RESTful APIs for programmatic access to the data and the implementation of automated workflows. InterMine has been successfully leveraged to build warehouses describing omics data from numerous organisms

Pathway Tools (v16.0)
Pathway Tools is a systems biology suite that may be used to build organism-specific databases—or model-organism databases (MODs)—that integrate various omics data types, from genomes to metabolic pathways. Those databases may be visualized and published on the web. Pathway Tools includes a variety of predictors for operons, transport reactions and the complete metabolic network of an organism, including missing enzymes leveraging the 1800+ manually curated pathways from MetaCyc as a reference. The suite also features numerous visualization tools , such as genome and pathway browsers, to facilitate the analysis and comparison of complete genomic data and metabolic networks, and includes Pathway/Genome Editors to facilitate the manual curation of MODs in a collaborative environment. It is also possible to share and/or download MODs that were made available by the community in the BioCyc database collection that now comprises nearly 3000 single organism databases.

3.7 Check Your Progress Questions

1. Where we are using Data warehousing?

3.8 Answers to Check Your Progress Questions

- Government
- Tourism
- Industry
- Genomics

3.9 Summary

Data warehouses are centralized data storage systems that allow your business to integrate data from multiple applications and sources into one location. This provides an environment that is designed for decision support, analytics reporting, and data mining. When you isolate and optimize your data, you can manage it without impacting primary business processes. In general, the benefits of data warehousing are all based on one central premise: warehousing solves the ongoing problem of analyzing separate data and converting it into actionable information you can use. Warehousing also allows you to process large amounts of complex data in an efficient way. When you successfully implement a data warehouse system, it's possible to access the benefits associated with the practice the very benefits that are making data warehousing a common practice for many businesses today. The following are some of the ways to increase efficiency, profitability and overall success through ETL and data warehousing.

3.10 Key Words

ETL : It is a type of data integration that refers to the three steps (extract, transform, load) used to blend data from multiple sources

3.11 Self-Assessment Questions and Exercises

1. Prepare case study about Data warehousing in Information Technology

3.12 Further Readings

1. Pang-Ning Tan, Michael Steinbach and Vipin Kumar, "Introduction to Data Mining", Person Education, 2007.
2. K.P. Soman, Shyam Diwakar and V. Aja, "Insight into Data Mining Theory and Practice", Eastern Economy Edition, Prentice Hall of India, 2006.
3. G. K. Gupta, "Introduction to Data Mining with Case Studies", Eastern Economy Edition, Prentice Hall of India, 2006.
4. Daniel T.Larose, "Data Mining Methods and Models", Wiley Interscience, 2006.
5. Alex Berson and Stephen J.Smith, "Data Warehousing, Data Mining and OLAP", Tata McGraw – Hill Edition, Thirteenth Reprint 2008.
6. Jiawei Han and Micheline Kamber, "Data Mining Concepts and Techniques", Third Edition, Elsevier, 2012.
7. A Review Of Genomic Data Warehousing Systems Briefings In Bioinformatics. Vol 15. No 4. 471^ 483 Doi:10.1093/Bib/Bbt031 Advance Access Published On 14 May 2013
8. Some Aspects Of Data Warehousing In Tourism IndustryThe Annals of The "Ştefan cel Mare" University Suceava. Fascicle of The Faculty of Economics and Public Administration Volume 9,No.1(9), 2009

BLOCK – 2 DATA MINING

UNIT – 4
INTRODUCTION TO DATA MINING

Structure

4.1 Introduction

Data Mining is the non-trivial process of identifying valid novel potentially useful and ultimately understandable patterns in data. With the widespread use of databases and the explosive growth in their sizes, organizations face the problem of information overload. Data mining techniques supports the automatic exploration of data and attempt to source out patterns and trends in the data and also infers rules from these patterns which will help the user to support a review and examine decisions in some related business or scientific area.

4.2 Objective

After going through the unit you will be able to:

- Understand the steps involved in KDD process.
- Know about architecture of typical data mining system
- Learn techniques used in data mining.
- Gain knowledge about current trends in data mining.

4.3 Definition

Data mining refers to extracting or mining knowledge from large amounts of data. It is the computational process of discovering patterns in large data sets involving methods at the intersection of artificial intelligence, machine learning, statistics, and database systems. The overall goal of the data mining process is to extract information from a data set and transform it into an understandable structure for further use.

The key properties of data mining are:

- ✓ Automatic discovery of patterns
- ✓ Prediction of likely outcomes
- ✓ Creation of actionable information
- ✓ Focus on large datasets and database

4.3.1 Knowledge Discovery in Databases (KDD)

Knowledge Discovery in Databases (KDD) is the process of identifying a valid, potentially useful and ultimately understandable structure in data. This process involves selecting or sampling data datawarehouse, cleaning or preprocessing it, transforming or reducing it, applying a data mining component to produce a structure, and then evaluating the derived structure. The following diagram shows the process of knowledge discovery process.

Data mining is a step in the KDD process.

Some people treat data mining same as Knowledge discovery while some people view data mining essential step in process of knowledge discovery. Here is the list of steps involved in knowledge discovery process:

Data Cleaning: It is the process of removing noise and inconsistent data.

Data Integration: It is the process of combining data from multiple sources.

Data Selection: It is the process of retrieving relevant data from the database.

Data Transformation: In this process, data are transformed or consolidated into forms appropriate for mining by performing summary or aggregation operations.

Data Mining: It is an essential process where intelligent methods are applied in order to extract data patterns.

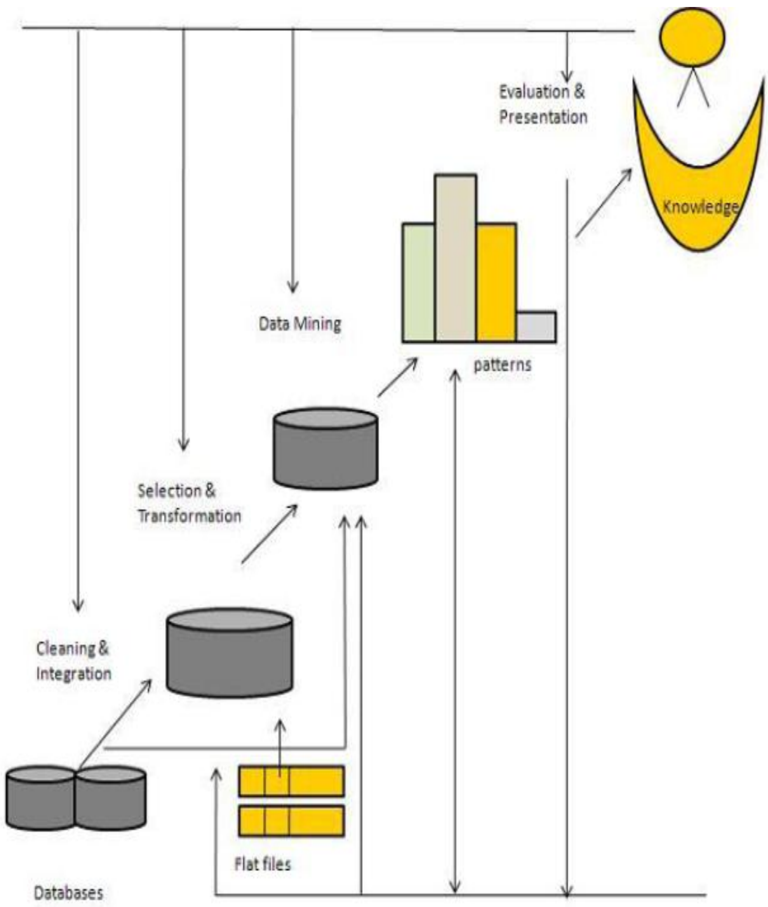


Figure 4.1 Steps in Knowledge Discovery.

Pattern Evaluation:

The patterns obtained in the data mining stage are converted into knowledge based on some interestingness measures.

Knowledge Presentation:

Visualization and knowledge representation techniques are used to

present the mined knowledge to the user.

4.3.2 Architecture of Data Mining :

Data Mining is the process of discovering interesting knowledge

from large amounts of data Stored either in databases, data warehouses or other information repositories. Based on this view,

the architecture of a typical data mining system may have the following major components.

Database, Data warehouse or other information repository: This is a single or a collection of multiple databases, data warehouses, flat files, spreadsheets or other kind of information repositories. Data cleaning and data integration techniques may be performed on the data.

Database or Data warehouse server: The database or data warehouse server fetches the relevant data, based on the user's data mining request.

Knowledge Base: This is the domain knowledge that is used to guide the search or evaluate the interestingness of resulting patterns. Such knowledge can include concept hierarchies, used to organize attributes or attribute values into different levels of abstraction knowledge such as user beliefs, which can be used to assess a pattern's interestingness.

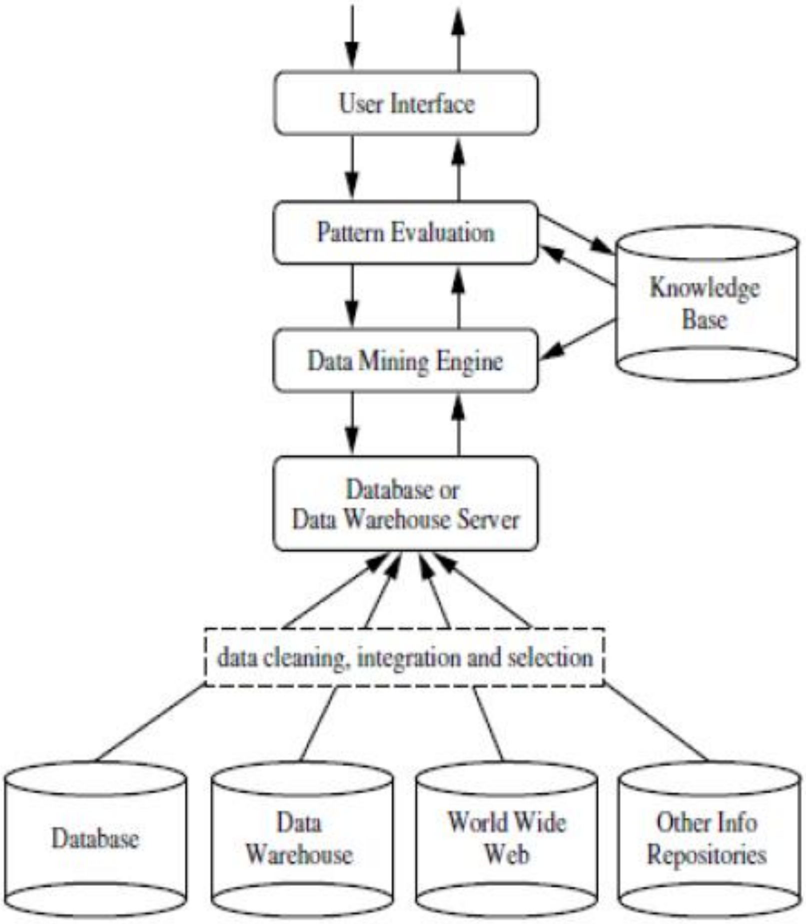


Figure 4.2 Architecture of Data Mining.

Data Mining Engine: This is essential to the data mining system and ideally consists of a set of functional modules for tasks such as characterization, association and correlation analysis, classification, prediction, cluster analysis, outlier analysis, and evolution analysis.

Pattern Evaluation Module: This component typically employs interestingness measures and interacts with the data mining modules to focus the search toward interesting patterns.

User interface: This module communicates between users and the data mining system, allowing the user to interact with the system by specifying a data mining query or task. Besides, this component allows the user to browse the database, evaluate mined patterns, and visualize the patterns in different forms.

4.4 Techniques in Data Mining

4.4.1 Association Rule

Association is one of the best-known data mining techniques. In association, a pattern is discovered based on a relationship between items in the same transaction. The association technique is used in *market basket analysis* to identify a set of products that customers frequently purchase together.

For example, retailers are using association techniques to research customer's buying habits. Based on historical sale data, retailers might find out that customers always buy jam when they buy bread, and, therefore, they can place bread and jam next to each other to save time for the customer and increase sales.

4.4.2 Classification

Classification is used to classify each item in a set of data into one of a predefined set of classes or groups. The classification method makes use of mathematical techniques such as decision trees, linear programming, neural network, and statistics. For example, we can apply classification in the application that "given all records of employees who left the company; predict who will probably leave the company in a future period". In this case, we divide the records of employees into two groups named “leave” and “stay”. And then we can ask our data mining software to classify the employees into separate groups.

4.4.3 Cluster Analysis

Clustering is a data mining technique that makes a meaningful or useful cluster of objects which have similar characteristics using the automatic technique.

For example in the library, we can keep books that have some kinds of similarities in one cluster or one shelf and label it with a meaningful name. If readers want to grab books on that topic, they would only have to go to that shelf instead of looking for the entire library.

4.4.4 Decision trees

Decision trees are simple knowledge representation and they classify examples/records to a finite numbers of classes, the node are labeled with attribute names, the edges are labeled with possible values for this attribute and the leaves labeled with different classes. A tree-shaped structure represents sets of decisions.

For example, we use the following decision tree to determine whether or not to play tennis:

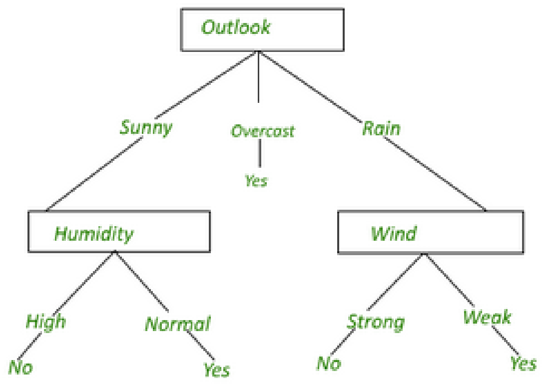


Figure 4.3 Decision Tree to play tennis.

4.4.5 Neural Network

Neural Networks have the remarkable ability to derive meaning from complicated or imprecise data and can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. Neural networks use a set of processing elements(or nodes) analogous to neurons in the brain. These processing elements are interconnected in a network that can then identify patterns in data once it is exposed to the data. Each node in the hidden layer is fully connected to the input which means that what is learned in a hidden node is based on all the inputs taken together.

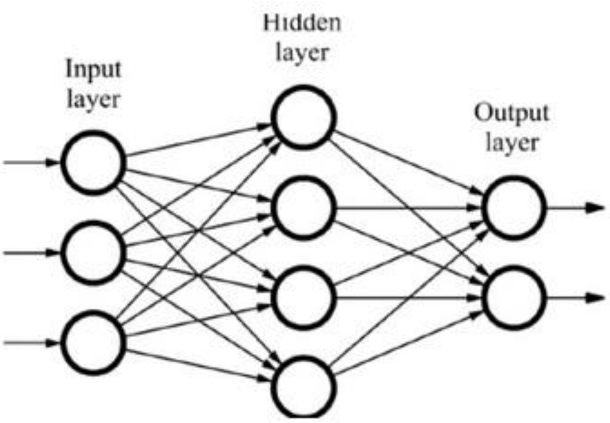


Figure 4.4 Structure of a neural network.

4.4.6 Prediction

The prediction, as its name implied, is one of the data mining techniques that discover the relationship between independent variables and the relationship between dependent and independent variables.
For example, the prediction analysis technique can be used in the sale to predict profit for the future if we consider the sale is an independent variable, profit could be a dependent variable. Then based on the historical sale and profit data, we can draw a fitted regression curve that is used for profit prediction.

4.5 Current Trends in Data Mining

The data may be incomplete and/or inaccurate. At times there may be redundant information, and at times there may not be sufficient information. It is also desirable to have data mining tools that can switch to multiple techniques and support multiple outcomes. Some of the current trends in data mining include the following and are illustrated in the Figure 4.5.

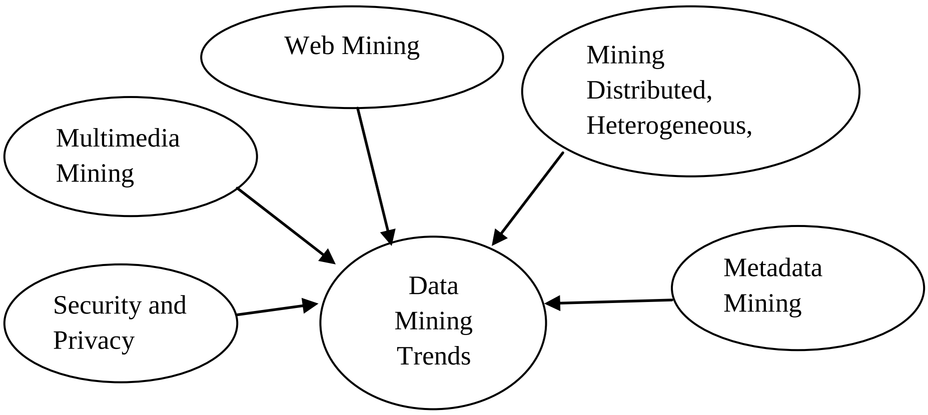


Figure 4.5 Trends in Data Mining.

- Mining distributed, heterogeneous, and legacy databases
- Mining multimedia data
- Mining data on the World Wide Web
- Security and privacy issues in data mining
- Metadata aspects of mining
- Application Exploration.
- Scalable and interactive data mining methods.
- Integration of data mining with database systems, data warehouse systems and web database systems.
- Standardization of data mining query language.
- Visual data mining.
- New methods for mining complex types of data.
- Biological data mining.
- Data mining and software engineering.
- Web mining.
- Distributed data mining.
- Real time data mining.
- Multi database data mining.
- Privacy protection and information security in data mining.

4.6 Check your progress questions

1. Define data mining?
2. Define KDD?
3. Mention the steps in KDD.

4.7 Answer to check your progress questions

1. Data mining refers to extracting or mining knowledge from large amounts of data.
2. Knowledge Discovery in Databases is the process involves selecting or sampling data from a data warehouse, cleaning or preprocessing it, transforming or reducing it, applying a data mining component to produce a structure, and then evaluating the derived structure.
3. Data cleaning, data integration, data selection , data transformation, data mining, pattern evaluation, and knowledge presentation are the steps in KDD.

4.8 Summary

Data mining is the process of discovering interesting patterns from massive amounts of data. As a knowledge discovery process, it typically involves data cleaning, data integration, data selection, data transformation, pattern discovery, pattern evaluation, and knowledge presentation. There are many data mining techniques only few has been discussed namely association rule, classification, cluster analysis, decision trees, neural network and prediction.

4.9 Keywords

- **Data Mining -** It is the process of discovering interesting knowledge from large amounts of data stored either in databases, data warehouses or other information repositories.
- **Data Cleaning -** It is the process of removing noise and inconsistent data.
- **Data Integration -** It is the process of combining data from multiple sources.
- **Data Selection -** It is the process of retrieving relevant data

  from the database.

- **Data Transformation -** In this process, data are transformed or consolidated into forms appropriate for mining by performing summary or aggregation operations.
- **Pattern Evaluation -** The patterns obtained in the data mining stage are converted into knowledge based on some interestingness measures.
- **Knowledge Presentation -** Visualization and knowledge representation techniques are used to present the mined knowledge to the user.

4.10 Self Assessment Questions and Exercises

1. Differentiate between data mining and KDD.
2. Describe the architecture of typical data mining system with a neat sketch.
3. Explain the steps of KDD with a neat sketch.
4. Discuss the various data mining techniques.
5. List out current trends in data mining.

4.11 Further Reading

1. Poonkuzhali. S, Saravanakumar. C, Data Warehousing & Data Mining, Charulatha Publications.
2. Jiawei Han, Micheline Kambar, Jian Pei, Data mining concepts and techniques, Morgan Kaufmann is an imprint of Elsevier.
3. Bhavani Thuraishingham (1999), Data Mining: Technologies, Techniques, Tools, and Trends, CRC Press LLC.
4. Bharat Bhushan Agarwal, Sumit Prakash Tayal, Data Mining and Data Warehousing, University Science Press.
5. Pang-Ning Tan, Vipin Kumar, Michael Steinbach, Introduction to Data Mining, Pearson.
6. Rao. N. Raghavendra, Global Virtual Enterprises in Cloud Computing Environments, United States of America by IGI Global.
7. https://www.datasciencecentral.com/profiles/blogs/the-7-most-important-data-mining-techniques.
8. https://www.zentut.com/data-mining/data-mining-techniques.

UNIT – 5

DIFFERENT FORMS OF KNOWLEDGE

Structure

5.1 Introduction

At the first international conference on KDD in Montreal in the year 1995, it was proposed that the term "KDD" be employed to describe the whole process of extraction of knowledge from data. In

this context knowledge means relationships and patterns between data and elements. It was proposed that the term "Data Mining" should be used exclusively for the discovery stage of the KDD process. It is clear that KDD is not an activity that stands on its own.

5.2 Objective

After going through the unit you will be able to:

- Understand different forms of knowledge
- Gain knowledge about six stages of KDD in detail.

5.3 Different Forms of Knowledge

The key issue in KDD is to realize that there is more information hidden in our data than we are able to distinguish four at first sight. In fact, in data mining we distinguish different types of knowledge that can be extracted from the data:

5.3.1 Shallow Knowledge

It is a piece of information that can be easily retrieved from databases using a query tool such as structured query language.

5.3.2 Multi-Dimensional Knowledge

It is a piece of information that can be analyzed using online analytical processing tools. With OLAP tools, we can rapidly explore all sorts of clustering, and different ordering of the data. But it is important to realize that, most of the things we can do with an OLAP tool can also be done using SQL. However, OLAP is not as powerful as data mining because it cannot search for optimal solutions.

5.3.3 Hidden Knowledge

This is data that can be found relatively easily by using pattern recognition or machine learning algorithms. Again one could use SQL to find these patterns but this would be extremely time-consuming. A pattern recognition algorithm could find regularities in a database in minutes or at most a couple of hours.

5.3.4 Deep Knowledge

Hidden knowledge is the result of a search space over a gentle, hilly landscape; a search algorithm can easily find an optimal solution. Deep knowledge is the result of a search space over only a tiny local optimum, with

no indication of any elevations in the neighborhood.

An example is an encrypted information stored in a database.

5.4 Data Selection

The knowledge discovery process consists of six stages: Data selection, Cleaning, Enrichment, Coding, Data mining, and Reporting.

In our example, we start by selecting a rough database containing

records of subscription data for the magazines. It is a selection of operational data from the publisher's invoicing system and contains information about people who have subscribed to a magazine.

The records consist of client number, name, address, date of subscription, and type of magazine. An illustration of the contents

of this database is given in Table 5.1.

Client Number	Name	Address	Data Purchase made	Mag Purc
23003	Johnson	1 Downing Street	04-15-94	C
23003	Johnson	1 Downing Street	06-21-93	Mı
23003	Johnson	1 Downing Street	05-30-92	Co
23009	Clinton	2 Boulevard	01-01-01	Co
23013	King	3 High Road	02-30-95	Sp
23019	Jonson	1 Downing Street	01-01-01	Ho

Table 5.1 Original Data.

5.5 Cleaning

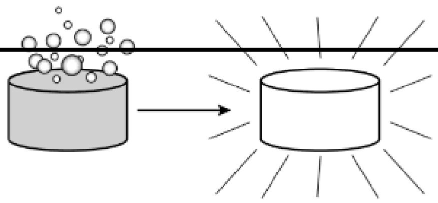


Figure 5.1 Data Cleaning.

A very important element in a cleaning operation is the de-duplication of records (see Table 5.2). In a normal client database some clients will be represented by several records, although in many cases this will be the result of negligence, such as people making typing errors, or of clients moving from one place to another without notifying change of address, spell their name incorrectly or give incorrect information about themselves, etc. Any company needs to be aware of such abnormalities in the database. Although data mining and data cleaning are two different disciplines, they have a lot in common and pattern recognition algorithms can be applied in cleaning data.

Client Number	Name	Address	Data Purchase made	Magazine Purchased
23003	Johnson	1 Downing Street	04-15-94	Car
23003	Johnson	1 Downing Street	06-21-93	Music
23003	Johnson	1 Downing Street	05-30-92	Comic
23009	Clinton	2 Boulevard	**01-01-01**	Comic
23013	King	3 High Road	02-30-95	Sports
23003	Johnson	1 Downing Street	**01-01-01**	House

Table 5.2 De-duplication.

A de-duplication algorithm using pattern analysis techniques could identify an error in the original database and correct it. After de-duplication, the two subscriptions of Mr. Johnson/Jonson can be recognized as those of one individual.

Next data cleaning, is the lack of domain consistency (see Table 5.3). Note that in our original table we have two records dated 1 January 1901, this is replaced with NULL values and corrected other domain inconsistencies.

Client Number	Name	Address	Data Purchase made	Magazine Purchased
23003	Johnson	1 Downing Street	04-15-94	Car
23003	Johnson	1 Downing Street	06-21-93	Music
23003	Johnson	1 Downing Street	05-30-92	Comic
23009	Clinton	2 Boulevard	NULL	Comic
23013	King	3 High Road	02-30-95	Sports
23003	Johnson	1 Downing Street	12-20-94	House

Table 5.3 Domain consistency.

5.6 Data Integration

Data integration combines data from multiple databases into a single database. During data integration, one has to detect and resolve data errors. Errors might be due to different values from different sources, different attribute formats, or attribute names might be different in different databases. One has to handle these kinds of redundant data to make sure the final database after integration consists of quality data. Data Integration reduces redundancies and inconsistencies and improves the speed of mining and produces quality information.

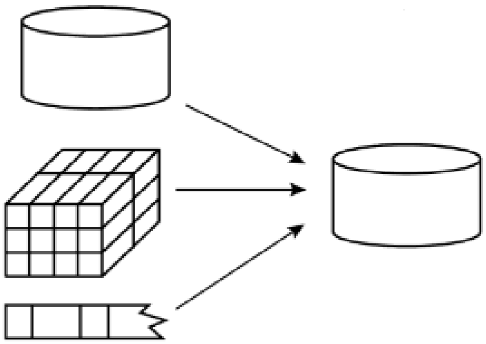


Figure 5.2 Data Integration.

5.7 Data Transformation

Data transformation converts the data into appropriate forms for mining. Smoothes the data summarizes and generalizes the data and constructs new attributes from the given ones. Normalization is done using min-max normalization, or z-normalization or by decimal scaling. For example attribute, data may be normalized to fall between a small range, such as 0.0 to 1.0.

−2, 32, 100, 59, 48 ⟶ −0.02, 0.32, 1.00, 0.59, 0.48

Figure 5.3 Data Transformation.

5.8 Data Reduction

Data reduction reduces the data set and provides a smaller volume data set, which yields similar results as the complete data sets.

Data reduction strategies include: *data aggregation* (e.g., building a data cube), *attribute subset selection* (e.g., removing irrelevant attributes through correlation analysis), *dimensionality reduction* (e.g., using encoding scheme such as minimum length encoding or wavelets), and *numerosity reduction* (e.g., “replacing” the data by alternative, smaller representations such as clusters or parametric models).

Categorical features can be reduced by using higher-level concepts. For example, attribute city substituted by country or state (higher-level concepts).

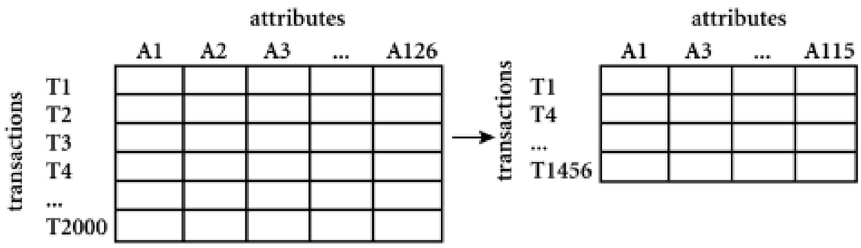


Figure 5.4 Data Transformation.

5.9 Enrichment

In the present example, we have purchased extra information about our clients consisting of the date of birth, income, amount of credit, and whether or not an individual owns a car or a house (Table 5.4). For this example, it is not important how the information was gathered, but it is necessary to appreciate that the new information can easily be joined to the existing client records.

Client name	*Date of birth*	*Income*	*Credit*	*Car owner*	*House owner*
Johnson	04-13-76	$18,500	$17,800	no	no
Clinton	10-21-71	$36,000	$26,600	yes	no

Table 5.4 Enrichment.

5.10 Check your progress questions

1. What are the different forms of knowledge?
2. Define data reduction.
3. Define data enrichment.

5.11 Answer to check your progress questions

1. Shallow knowledge, Multi-Dimensional knowledge, Hidden knowledge and Deep knowledge are the different types of knowledge.
2. Data reduction reduces the data set and provides a smaller volume data set, which yields similar results as the complete data sets.
3. Data enrichment is necessary stage KDD, where the new information can easily be joined to the existing client records.

5.12 Summary

The knowledge discovery process consists of six stages: Data selection, Cleaning, Enrichment, Coding, Data mining, and Reporting.

Data cleaning routines attempt to fill in missing values, smooth out noise while identifying outliers, and correct inconsistencies in the data. Data integration combines data from multiple sources to form a coherent data store. Data reduction techniques obtain a reduced representation of the data while minimizing the loss of information content. Data transformation converts the data into appropriate forms for mining.

5.13 Keywords

Shallow Knowledge - It is a piece of information that can be easily retrieved from databases using a query tool such as structured query language.

Multi-Dimensional Knowledge - It is a piece of information that can be analyzed using online analytical processing tools.
Hidden Knowledge **-** This is data that can be found relatively easily by using pattern recognition or machine learning algorithms.
Deep Knowledge **-** Deep knowledge is the result of a search space over only a tiny local optimum, with no indication of any elevations in the neighborhood.

5.14 Self Assessment Questions and Exercises

1. Explain about different forms of knowledge
2. Briefly explain about a) Data Selection b) Data Cleaning c) Data Integration d) Data transformation e) Data reduction and f) Data enrichment.

5.15 Further Reading

1. Poonkuzhali. S, Saravanakumar. C, Data Warehousing & Data Mining, Charulatha Publications.
2. Jiawei Han, Micheline Kambar, Jian Pei, Data mining concepts and techniques, Morgan Kaufmann is an imprint of Elsevier.
3. Bharat Bhushan Agarwal, Sumit Prakash Tayal, Data Mining and Data Warehousing, University Science Press
4. Pang-Ning Tan, Vipin Kumar, Michael Steinbach, Introduction to Data Mining, Pearson

UNIT – 6
DATA

Structure

6.1 Introduction

Let us discuss several data-related issues that are important for successful data mining: The Type of Data, The Quality of the Data, Preprocessing Steps to Make the Data More suitable for Data Mining and Analyzing Data in Terms of Its Relationships.

6.2 Objective

After going through the unit you will be able to:

- Understand the meaning of attributes, data sets, types of attributes and types of data sets.
- Learn in detail about data quality and data preprocessing.
- Discuss different strategies and techniques interrelated in data preprocessing.
- Also gain knowledge on measures of similarity and dissimilarity between attributes and data objects.

6.3 Data

A data set can often be viewed as a collection of data objects. Other names for a data object are record, point, vector, pattern, event, case, sample, observation, or entity. In turn, data objects are described by a number of attributes that capture the basic characteristics of an object, such as the mass of a physical object or the time at which an event occurred. Other names for an attribute are variable, characteristic, field, feature, or dimension.

6.4 Types of Data

A data set can often be viewed as a collection of data objects. Other names for a data object are a record, point, vector, pattern, event, case, sample, observation, or entity. Data objects are described by several attributes that capture the basic characteristics of an object.

Student ID	Year	Grade Point Average (GPA)	...
	⋮		
1034262	Senior	3.24	...
1052663	Sophomore	3.51	...
1082246	Freshman	3.62	...
	⋮		

Table 6.1 A sample dataset containing student information.

6.4.1 Attribute and measurement

An attribute is a property or characteristic of an object that may vary; either from one object to another or from one time to another. A measurement scale is a rule (function) that associates a numerical or symbolic value with an attribute of an object.

The Different Type of an Attribute

The four types of attributes are nominal, ordinal, interval, and ratio. Table 6.2 gives the definitions of these types, along with information about the statistical operations that are valid for each type. Each attribute type possesses all of the properties and operations of the attribute types above it. Nominal and ordinal attributes are collectively referred to as categorical or qualitative attributes. The remaining two types of attributes, interval, and ratio are collectively referred to as quantitative or numeric attributes.

Attribute Type		Description	Examples	Operations
Categorical (Qualitative)	Nominal	The values of a nominal attribute are just different names; i.e., nominal values provide only enough information to distinguish one object from another. ($=, \neq$)	zip codes, employee ID numbers, eye color, gender	mode, entropy, contingency correlation, χ^2 test
	Ordinal	The values of an ordinal attribute provide enough information to order objects. ($<, >$)	hardness of minerals, $\{good, better, best\}$, grades, street numbers	median, percentiles, rank correlation, run tests, sign tests
Numeric (Quantitative)	Interval	For interval attributes, the differences between values are meaningful, i.e., a unit of measurement exists. ($+, -$)	calendar dates, temperature in Celsius or Fahrenheit	mean, standard deviation, Pearson's correlation, t and F tests
	Ratio	For ratio variables, both differences and ratios are meaningful. ($*, /$)	temperature in Kelvin, monetary quantities, counts, age, mass, length, electrical current	geometric mean, harmonic mean, percent variation

Table 6.2 Different attribute types.

Describing Attributes by the Number of Values

An independent way of distinguishing between attributes is by the number of values they can take, namely discrete and continuous.

A discrete attribute has a finite or countably infinite set of values. Such attributes can be categorical, such as zip codes or ID numbers, or numeric, such as counts. Binary attributes are a special case of discrete attributes and assume only two values, e.g., true/false, male/female, or 0/1.A continuous attribute is one whose values are real numbers. Examples include attributes such as temperature, height, or weight.

Asymmetric Attributes

In asymmetric attributes, only the presence of the value of a non-zero attribute is regarded as important. Binary attributes where only non-zero values are important are called asymmetric binary attributes.

6.3.2 Types of Data Sets

There are many types of data sets. For convenience, we have grouped the types of data sets into three groups: record data, graph-based data, and ordered data.

General Characteristics of Data Sets

The three characteristics that apply to many data sets and have a significant impact on the data mining techniques are dimensionality, sparsity, and resolution.

Dimensionality

The dimensionality of a data set is the number of attributes that the objects in the data set possess. Data with a small number of dimensions tend to be qualitatively different than moderate or high-dimensional data. Indeed, the difficulties associated with analyzing high-dimensional data are sometimes referred to as the curse of dimensionality. Because of this, an important motivation in preprocessing the data is dimensionality reduction.

Sparsity

For some data sets, such as those with asymmetric features, most attributes of an object have values of 0. In many cases less than 1% of the entries are non-zero. Sparsity is an advantage because of the results which have significant savings for computation time and storage.

Resolution

It is frequently possible to obtain data at different levels of resolution, and often the properties of the data are different at different resolutions. For example, the surface of the Earth seems very uneven at a resolution of a few meters but is relatively smooth at a resolution of tens of kilometers. The patterns in the data also depend on the level of resolution. If the resolution is too fine, a pattern may not be visible or may be buried in noise;

Tid	Refund	Marital Status	Taxabke Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

(a) Record data

Tid	ITEMS
1	Bread, Soda, Milk
2	Beer, Bread
3	Deer, Soda, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Soda, Diaper, Milk

(b) Transaction data

Projection of x Load	Projection of y Load	Distance	Load	Thickness
10.23	5.27	15.22	27	1.2
12.65	6.25	16.22	22	1.1
13.54	7.23	17.34	23	1.2
14.27	8.43	18.45	25	0.9

(c) Data matrix

	team	coach	play	ball	score	game	win	lost	timeout	season
Document 1	3	0	5	0	2	6	0	2	0	2
Document 2	0	7	0	2	1	0	0	3	0	0
Document 3	0	1	0	0	1	2	2	0	3	0

(d) Document-term matrix

Figure 6.1 Different variations of record data

Record Data

Much data mining work assumes that the data set is a collection of records (data objects), each of which consists of a fixed set of data fields (attributes). See Figure 6.1(a).

Transaction or Market Basket Data

Transaction data is a special type of record data, where each record (transaction) involves a set of items. Consider a grocery store. The set of products purchased by a customer during one shopping trip constitutes a transaction, while the individual products that were purchased are the items. This type of data is called market basket data.

The Data Matrix

A set of such data objects can be interpreted as an m by n matrix, where there are m rows, one for each object, and n columns, one for each attribute. (A representation that has data objects as columns and attributes as rows is also fine.) This matrix is called a data matrix or a pattern matrix.

The Sparse Data Matrix

A sparse data matrix is a special case of a data matrix in which the attributes are of the same type and are asymmetric; i.e., only non-zero values are important. Figure 6.1(c) shows a sample data

matrix.Another common example is document data. In particular, if the order of the terms (words) in a document is ignored, then a document can be represented as a term vector, where each term is a component (attribute) of the vector and the value of each component is the number of times the corresponding term occurs in the document. This representation of a collection of documents is often called a document-term matrix. Figure 2.9(d) shows a sample document-term matrix.

Graph-Based Data

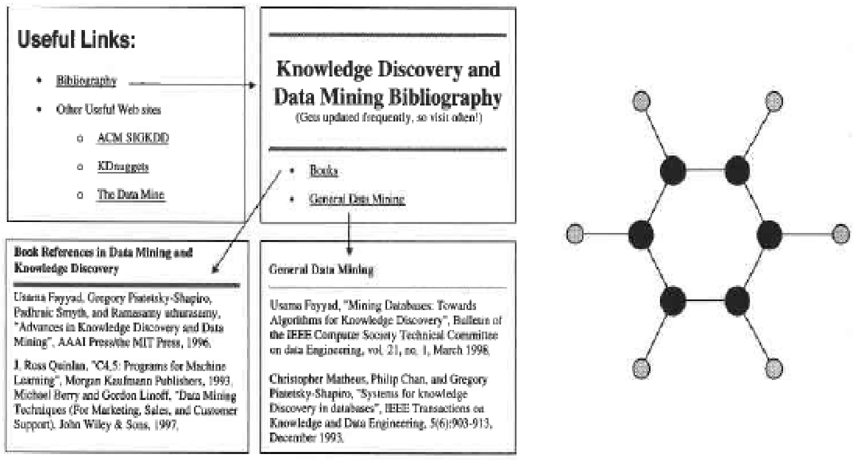
A graph can sometimes be a convenient and powerful representation of data. We consider two specific cases: (1) the graph captures relationships among data objects and (2) the data objects themselves are represented as graphs.

Data with Relationships among Objects

The relationships among objects frequently convey important information. In such cases, the data is often represented as a graph. In particular, the data objects are mapped to nodes of the graph, while the relationships among objects are captured by the links between objects and link properties, such as direction and weight. Consider Web pages on the World Wide Web, which contain both text and links to other pages.

Data with Objects That Are Graphs

If objects have structure, that is, the objects contain sub-objects that have relationships, then such objects are frequently represented as graphs. For example, the structure of chemical compounds can be represented by a graph, where the nodes are atoms and the links between nodes are chemical bonds. Figure 6.2(b) shows a ball-and-stick diagram of the chemical compound benzene, which contains atoms of carbon (black) and hydrogen (gray).



(a) Linked Web pages. (b) Benzene molecule.

Figure 6.2 Different variations of graph data

Ordered Data

For some types of data, the attributes have relationships that involve order in time or space.

Sequential Data

Sequential data also referred to as temporal data, can be thought of as an extension of record data, where each record has a time associated with it. Consider a retail transaction data set that also stores the time at which the transaction took place.

Sequence Data

Sequence data consists of a data set that is a sequence of individual entities, such as a sequence of words or letters. It is quite similar to sequential data, except that there are no time stamps; instead, there are positions in an ordered sequence. For example, the genetic information of plants and animals can be represented in the form of sequences of nucleotides that are known as genes.

Time Series

Data Time series data is a special type of sequential data in which each record is a time series, i.e., a series of measurements taken over time. For example, a financial data set might contain objects that are time series of the daily prices of various stocks.

Spatial Data

Some objects have spatial attributes, such as positions or areas, as well as other types of attributes. An example of spatial data is weather data (precipitation, temperature, pressure) that is collected for a variety of geographical locations.

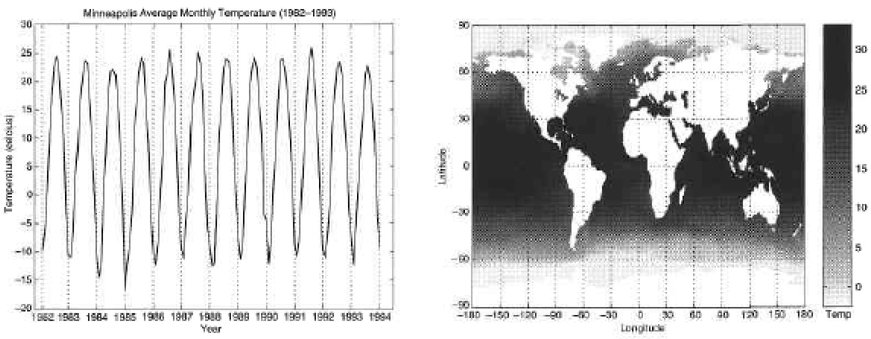
Time	Customer	Items Purchased
t1	C1	A, B
t2	C3	A, C
t2	C1	C, D
t3	C2	A, D
t4	C2	E
t5	C1	A, E

Customer	Time and Items Purchased
C1	(t1: A,B) (t2:C,D) (t5:A,E)
C2	(t3: A, D) (t4: E)
C3	(t2: A, C)

(a) Sequential transaction data.

GGTTCCGCCTTCAGCCCCGCGCC
CGCAGGGCCCGCCCCGCGCCGTC
GAGAAGGGCCCGCCTGGCGGGCG
GGGGGAGGCGGGGCCGCCCGAGC
CCAACCGAGTCCGACCAGGTGCC
CCCTCTGCTCGGCCTAGACCTGA
GCTCATTAGGCGGCAGCGGACAG
GCCAAGTAGAACACGCGAAGCGC
TGGGCTGCCTGCTGCGACCAGGG

(b) Genomic sequence data.



(c) Temperature time series. (d) Spatial temperature data.

Figure 6.3 Different variations of ordered data

6.5 Data Quality

Data mining focuses on (1) the detection and correction of data quality problems and (2) the use of algorithms that can tolerate poor data quality. The first step, detection, and correction, is often called data cleaning.

6.5.1 Measurement and Data Collection Issues

It is unrealistic to expect that the data will be perfect. There may be problems due to human error, limitations of measuring devices, or flaws in the data collection process. In this section, let us focus on aspects of data quality that are related to data measurement and collection issues and issues related to applications.

Measurement and Data Collection Errors

The term measurement error refers to any problem resulting from the measurement process. A common problem is that the value recorded differs from the true value to some extent.

The term data collection error refers to errors such as omitting data objects or attribute values or inappropriately including a data object. Both measurement errors and data collection errors can be either systematic or random.

For example, keyboard errors are common when data is entered manually, and as a result, many data entry programs have techniques for detecting and, with human intervention, correcting such errors.

Noise and Artifacts

Noise is the random component of a measurement error. Figure 6.4 shows a time series before and after it has been disrupted by random noise. If a bit more noise were added to the time series, its shape would be lost. Figure 6.5 shows a set of data points before and after some noise points (indicated by '+'s) have been added.

Data errors may be the result of a more deterministic phenomenon, such as a streak in the same place on a set of photographs. Such deterministic distortions of the data are often referred to as artifacts.



(a) Time series. (b) Time series with noise.

Figure 6.4 Noise in a time series context

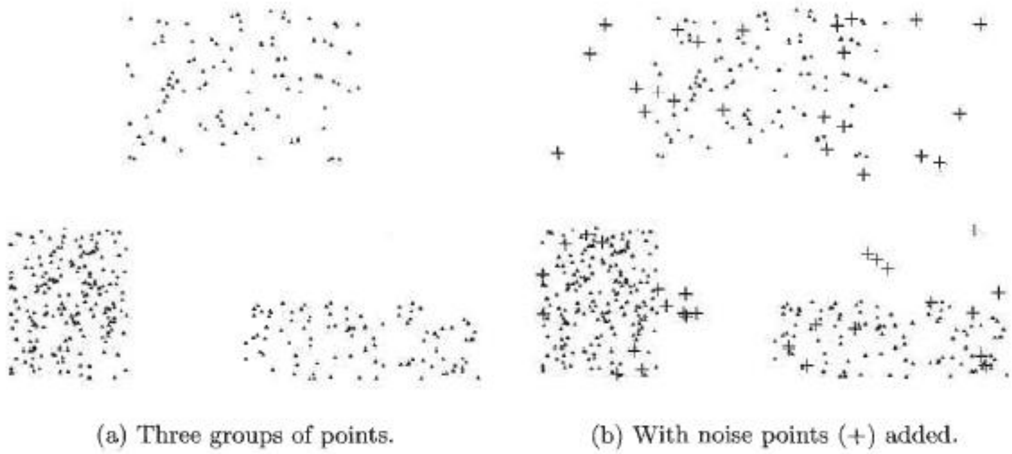


Figure 6.5 Noise in a spatial context.

Precision, Bias, and Accuracy
Precision is the closeness of repeated measurements (of the same quantity) to one another.

Bias is a systematic variation of measurements from the quantity being measured.

Accuracy is the closeness of measurements to the true value of the quantity being measured.

Precision is often measured by the standard deviation of a set of values, while bias is measured by taking the difference between the mean of the set of values and the known value of the quantity being measured. Accuracy depends on precision and bias, but since it is a general concept, there is no specific formula for accuracy in terms of these two quantities.

Outliers

Outliers are either (1) data objects that have characteristics that are different from most of the other data objects in the data set, or (2) values of an attribute that are unusual for the typical values for that attribute. Outliers can be legitimate data objects or values. Thus, unlike noise, outliers may sometimes be of interest. Examples are fraud and network intrusion detection.

Missing Values

It is not unusual for an object to be missing one or more attribute values. In some cases, the information was not collected; e.g., some people decline to give their age or weight.

Eliminate Data Objects or Attributes

A simple and effective strategy is to eliminate objects with missing values. However, even a partially specified data object contains some information, and if many objects have missing values, then a reliable analysis can be difficult or impossible.

Estimate Missing Values

Sometimes missing data can be reliably estimated. For example, consider a time series that changes in a reasonably smooth fashion, but has a few, widely scattered missing values. In such cases, the missing values can be estimated (interpolated) by using the remaining values.

Inconsistent Values

Data can contain inconsistent values. Consider an address field, where both a zip code and city are listed, but the specified zip code area is not contained in that city.

Regardless of the cause of the inconsistent values, it is important to detect and, if possible, correct such problems. Some types of inconsistencies are easy to detect. For instance, a person's height should not be negative. Once an inconsistency has been detected, it is sometimes possible to correct the data.

Duplicate Data

A data set may include data objects that are duplicates, or almost duplicates, of one another. Many people receive duplicate mailings because they appear in a database multiple times under slightly different names.

The duplicates are legitimate, but may still cause problems for some algorithms if the possibility of identical objects is not accounted for in their design. To detect and eliminate such duplicates, two main issues should be considered.

First, if two objects actually represent a single object, then the values of corresponding attributes may differ, and these inconsistent values must be resolved. Second, care needs to be taken to avoid accidentally combining similar data objects, but not duplicates, such as two distinct people with identical names. The term de- these issues.

6.5.2 Issues Related to Applications

Data quality has proven quite useful, particularly in business and industry. A similar viewpoint is also present in statistics and the experimental sciences, with their emphasis on the careful design of experiments to collect the data relevant to a specific hypothesis.

Timeliness

Some data starts to age as soon as it has been collected. In particular, if the data provides a snapshot of some ongoing phenomenon or process, such as the purchasing behavior of customers or Web browsing patterns, then this snapshot represents reality for only a limited time. If the data is out of date, then so are the models and patterns that are based on it.

Relevance

The available data must contain the information necessary for the application. Consider the task of building a model that predicts the accident rate for drivers. If information about the age and gender of the driver is omitted, then it is likely that the model will have limited accuracy unless this information is indirectly available through other attributes.

6.6 Data Pre-processing

Data pre-processing is a broad area and consists of many different strategies and techniques that are interrelated in complex ways. We will present some of the most important ideas and approaches, and try to point out the interrelationships among them.

- Aggregation
- Sampling
- Dimensionality reduction
- Feature subset selection
- Feature creation
- Discretization and binarization
- Variable transformation

These items fall into two categories namely selecting data objects and attributes for the analysis or creating/changing the attributes. In both cases, the goal is to improve data mining analysis with time, cost, and quality.

6.6.1 Aggregation

Aggregation is the combining of two or more objects into a single object. Consider a data set consisting of transactions (data objects) recording the daily sales of products in various store locations (Minneapolis, Chicago, Paris, etc.)

for different days over the year. See Table 6.3. One way to aggregate transactions for this data set is to replace all the transactions of a single store with a single storewide transaction. This reduces the hundreds or thousands of transactions that occur daily at a specific store to a single daily transaction, and the number of data objects is reduced to the number of stores.

Transaction ID	Item	Store Location	Date	Price	...
⋮	⋮	⋮	⋮	⋮	
101123	Watch	Chicago	09/06/04	$25.99	...
101123	Battery	Chicago	09/06/04	$5.99	...
101124	Shoes	Minneapolis	09/06/04	$75.00	...
⋮	⋮	⋮	⋮	⋮	

Table 6.3 Data set containing information about customer purchases

The data in Table 6.3 can also be viewed as a multidimensional array, where each attribute is a dimension. From this viewpoint, aggregation is the process of eliminating attributes, such as the type of item, or reducing the number of values for a particular attribute; e.g., reducing the possible values for a date from 365 days to 12 months. This type of aggregation is commonly used in Online Analytical Processing (OLAP).

6.6.2 Sampling

Sampling is a commonly used approach for selecting a subset of the data objects to be analyzed. The data miners sample because it is too expensive or time-consuming to process all the data. The representativeness of any particular sample will vary, and the best that we can do is choose a sampling scheme that guarantees a high probability of getting a representative sample.

Sampling Approaches

The simplest type of sampling is simple random sampling. There are two variations on random sampling (and other sampling techniques as well), (1) sampling without replacement, as each item is selected, it is removed from the set of all objects that together constitute the population, and (2) sampling with replacement, objects are not removed from the population as they are selected for the sample.

Progressive Sampling

The proper sample size can be difficult to determine, so adaptive or progressive sampling schemes are sometimes used. These approaches start with a small sample and then increase the sample

size until a sample of sufficient size has been obtained. Progressive sampling is used to learn a predictive model.

6.6.3 Dimensionality Reduction

Data sets can have a large number of features. Consider a set of documents, where each document is represented by a vector whose components are the frequencies with which each word occurs in the document. In such cases, there are typically thousands or tens of thousands of attributes (components), one for each word in the vocabulary. There are a variety of benefits to dimensionality reduction. A key benefit is that many data mining algorithms work better if the dimensionality of the number of attributes in the data is lower.

The Curse of Dimensionality

The curse of dimensionality refers to the phenomenon that many types of data analysis become significantly harder as the dimensionality of the data increases. Specifically, as dimensionality increases, the data becomes increasingly sparse in the space that it occupies. For classification, this can mean that there are not enough data objects to allow the creation of a model that reliably assigns a class to all possible objects.

Linear Algebra Techniques for Dimensionality Reduction

Some of the most common approaches for dimensionality reduction, particularly for continuous data, use techniques from linear algebra to project the data from a high-dimensional space into a lower-dimensional space. Principal Components Analysis (PCA) is a linear algebra technique for continuous attributes that finds new attributes (principal components) that (1) are linear combinations of the original attributes, (2) are orthogonal (perpendicular) to each other, and (3) capture the maximum amount of variation in the data.

6.6.4 Feature Subset Selection

Another way to reduce the dimensionality is to use only a subset of the features. While it might seem that such an approach would lose information, this is not the case if redundant and irrelevant features are present. Redundant features duplicate much or all of the information contained in one or more other attributes.

For example, the purchase price of a product and the amount of sales tax paid contain much of the same information. Irrelevant features contain almost no useful information for the data mining task at hand. There are three standard

approaches to feature selection: embedded, filter, and wrapper.

Embedded approaches

Feature selection occurs naturally as part of the data mining algorithm. Specifically, during the operation of the data mining algorithm, the algorithm itself decides which attributes to use and which to ignore.

Filter approaches

Features are selected before the data mining algorithm is run, using some approach that is independent of the data mining task. For example, we might select sets of attributes whose pairwise correlation is as low as possible.

Wrapper approaches

These methods use the target data mining algorithm as a black box to find the best subset of attributes, in a way similar to that of the ideal algorithm but typically without enumerating all possible subsets.

Architecture for Feature Subset Selection

The feature selection process is viewed as consisting of four parts: a measure for evaluating a subset, a search strategy that controls the generation of a new subset of features, a stopping criterion, and a validation procedure. Filter methods and wrapper methods differ only in the way in which they evaluate a subset of features. For a wrapper method, subset evaluation uses the target data mining algorithm, while for a filter approach, the evaluation technique is distinct from the target data mining algorithm.

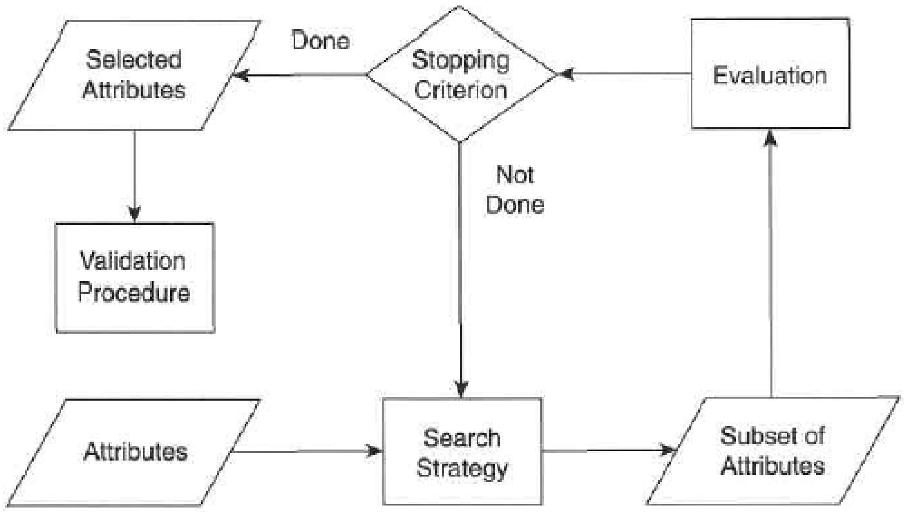


Figure 6.6 Flowchart of a feature subset selection process

Feature Weighting

Feature weighting is an alternative to keeping or eliminating features. More important features are assigned a higher weight, while less important features are given a lower weight. These weights are sometimes assigned based on domain knowledge about the relative importance of features.

6.6.5 Feature Creation

It is frequently possible to create, from the original attributes, a new set of attributes that captures the important information in a data set much more effectively. Furthermore, the number of new attributes can be smaller than the number of original attributes, allowing us to reap all the previously described benefits of dimensionality reduction.

Feature Extraction

The creation of a new set of features from the original raw data is known as feature extraction. Consider a set of photographs, where each photograph is to be classified according to whether or not it contains a human face. The raw data is a set of pixels, and as such, is not suitable for many types of classification algorithms. Whenever data mining is applied to a relatively new area, a key task is the development of new features and feature extraction methods.

Mapping the Data to a New Space

A different view of the data can reveal important and interesting features. Consider, for example, time-series data, which often contains periodic patterns. If there is only a single periodic pattern and not much noise' then the pattern is easily detected.

Feature Construction

Sometimes the features in the original data sets have the necessary information, but it is not in a form suitable for the data mining algorithm. In this situation, one or more new features constructed out of the original features can be more useful than the original features.

6.6.6 Discretization and Binarization

Some data mining algorithms, especially certain classification algorithms, require that the data be in the form of categorical attributes. Algorithms that find association patterns require that the data be in the form of binary attributes. Thus, it is often necessary to transform a continuous attribute into a categorical attribute (discretization), and both continuous and discrete attributes may need to be transformed into one or more binary attributes (binarization). As with feature selection, the best discretization and binarization approach is the one that "produces the best result for the data mining algorithm that will be used to analyze the data."

Binarization

A simple technique to binarize a categorical attribute is the following: If there are m categorical values, then uniquely assign each original value to an integer in the interval $[0, m-1]$. Next, convert each of these m integers to a binary number since $n = \lceil \log_2(m) \rceil$. To illustrate, a categorical variable with 5 values $\{awful, poor, OK, good, great\}$ would require three binary variables x1, x2 and x3. The conversion is shown in Table 6.4.

Categorical Value	Integer Value	x_1	x_2	x_3
awful	0	0	0	0
poor	1	0	0	1
OK	2	0	1	0
good	3	0	1	1
great	4	1	0	0

Table 6.4 Conversion of a categorical attribute to three binary attributes.

Discretization of Continuous Attributes

Discretization is typically applied to attributes that are used in classification or association analysis. Transformation of a continuous attribute to a categorical attribute involves two subtasks: deciding how many categories to have and determining how to map the values of the continuous attribute to these categories.

Unsupervised Discretization

A basic distinction between discretization methods for classification is whether class information is used (supervised) or not (unsupervised). For instance, the equal width approach divides the range of the attribute into a user-specified number of intervals each having the same width. Such an approach can be badly affected by outliers, and for that reason, an equal frequency (equal depth) approach, which tries to put the same number of objects into each interval, is often preferred.

Supervised Discretization

Figure 6.7(a) shows data points belonging to four different groups, along with two outliers, the large dots on either end. The split points produced by the techniques equal width, equal frequency, and K-means are shown in Figures 6.7(b), 6.7(c), and 6.7(d), respectively. The split points are represented as dashed lines.

The discretization methods described above are usually better than no discretization, but keeping the end purpose in mind and using additional information (class labels) often produces better results. Entropy based approaches are one of the most promising approaches to discretization.

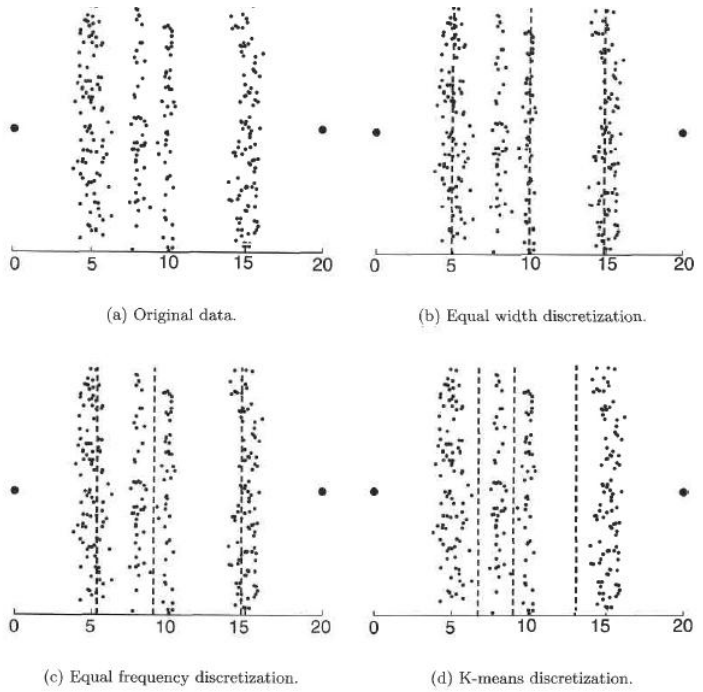


Figure 6.7 Different discretization techniques

6.6.7 Variable Transformation

A variable transformation refers to a transformation that is applied to all the values of a variable. For example, if only the magnitude of a variable is important, then the values of the variable can be transformed by taking the absolute value. Two important types of variable transformations: simple functional transformations and normalization.

Simple Functions

For this type of variable transformation, a simple mathematical function is applied to each value individually. If r is a variable, then examples of such transformations include x^k, $\log x$, e^x, $\sqrt{x}$, $1/x$, $\sin x$, or $|x|$.

Normalization or Standardization

Another common type of variable transformation is the standardization or normalization of a variable. The goal of standardization or normalization is to make an entire set of values have a particular property. A traditional example is that of "standardizing a variable" in statistics. If $\bar{x}$ is the mean (average) of

the attribute values and s_x is their standard deviation, then the transformation $x' = (x - \bar{x})/s_x$ creates a new variable that has a mean of 0 and a standard deviation of 1.

6.7 MEASURES OF SIMILARITY AND DISSIMLARITY

The term proximity is used to refer to either similarity or dissimilarity.

6.7.1 Basics

Definitions

The similarity between two objects is a numerical measure of the degree to which the two objects are alike. Similarities are usually non-negative and are often between 0 (no similarity) and 1 (complete similarity). Similarities are usually non-negative and are often between 0 (no similarity) and 1 (complete similarity). The dissimilarity between two objects is a numerical measure of the degree to which the two objects are different. Dissimilarities are lower for more similar pairs of objects. Dissimilarities sometimes fall in the interval [0,1], but it is also common for them to range from 0 to ∞.

Transformations

Transformations are often applied to convert a similarity to dissimilarity, or vice versa, or to transform a proximity measure to fall within a particular range, such as [0,1]. In the more general case, the transformation of similarities to the interval [0,1] is given by the expression $s' = (s - \min s)/(\max s - \min s)$, where $\max s$ and $\min s$ are the maximum and minimum similarity values, respectively. Likewise, dissimilarity measures with a finite range can be mapped to the interval [0, 1] by using the formula $d' = (d - \min d)/(\max d - \min d)$.

6.7.2 Similarity and Dissimilarity between Simple Attributes

Attribute Type	Dissimilarity	Similarity
Nominal	$d = \begin{cases} 0 & \text{if } x = y \\ 1 & \text{if } x \neq y \end{cases}$	$s = \begin{cases} 1 & \text{if } x = y \\ 0 & \text{if } x \neq y \end{cases}$
Ordinal	$d = \|x - y\|/(n-1)$ (values mapped to integers 0 to $n-1$, where n is the number of values)	$s = 1 - d$
Interval or Ratio	$d = \|x - y\|$	$s = -d$, $s = \frac{1}{1+d}$, $s = e^{-d}$, $s = 1 - \frac{d - min_d}{max_d - min_d}$

Table 6.5 Similarity and dissimilarity for simple attributes

Consider an attribute that measures the quality of a product, e.g., a candy bar, on the scale$\{poor, fair, OK, good, wonderful\}$. It would seem reasonable that a product, P1, which is rated wonderful, would be closer to a product P2, which is rated good, than it would be to a product P3, which is rated OK. To make this observation quantitative, the values of the ordinal attribute are often mapped to successive integers, beginning at 0 or 1, e.g., $\{poor = 0, fair = 1, OK = 2, good = 3, wonderful = 4\}$
Then, $d(P1, P2) = 3 - 2 = 1$ or, if we want the dissimilarity to fall between 0 and 1, $d(P1, P2) = \frac{3-2}{4} = 0.25$. A similarity for ordinal attributes can then be defined as $s = 1 - d$.

6.7.3 Dissimilarities between Data Objects

We begin with a discussion of distances, which are dissimilarities with certain properties, and then provide examples of more general kinds of dissimilarities.

Distances
The Euclidean distance, *d*, between two points, x and y, in one-, two-, three-, or higher- dimensional space, is given by the following familiar formula:

$$d(x, y) = \sqrt{\sum_{k=1}^{n} (x_k - y_k)^2} \qquad (6.1)$$

where n is the number of dimensions and x_k and y_k are, respectively, the k^{th} attributes (components) of x and y.

Distances, such as the Euclidean distance, have some well-known properties. If d(x, y) is the distance between two points, x and y, then the following properties hold.

1. **Positivity**
   (a) $d(x, x) \geq 0$ *for all x and y.*
   (b) $d(x, y) = 0$ *for all* $x = y$
2. **Symmetry**
   $d(x, y) = d(y, x)$ *for all x and y.*

3. **Triangle Inequality**
   $d(x, z) \leq d(x, y) + d(y, z)$ *for all points x, y and z.*

Measures that satisfy all three properties are known as metrics.

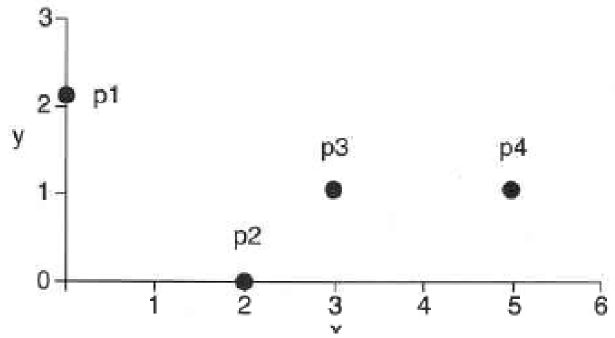


Figure 6.8 Four Two-Dimensional points

point	x coordinate	y coordinate
p1	0	2
p2	2	0
p3	3	1
p4	5	1

Table 6.6 x and y coordinates of four points

	p1	p2	p3	p4
p1	0.0	2.8	3.2	5.1
p2	2.8	0.0	1.4	3.2
p3	3.2	1.4	0.0	2.0
p4	5.1	3.2	2.0	0.0

Table 6.7 Euclidean distance matrix for Table 6.6

6.7.4 Similarities between Data Objects

For similarities, the triangle inequality typically does not hold, but symmetry and positivity typically do. To do explicit, if s(x,y) is the similarity between points x and y, then the typical properties of similarities are the following:

1. $s(x, y) = 1 \; only \; if \; x = y. (0 \leq s \leq 1)$
2. $s(x, y) = s(y, x) \; for \; all \; x \; and \; y. (Symmetry)$

There is no general analog of the triangle inequality for similarity measures. It is sometimes possible, however, to show that a similarity measure can easily be converted to a metric distance.

Similarity Measures for Binary Data

Similarity measures between objects that contain only binary attributes are called similarity coefficients, and typically have values between 0 and 1.Let x and y be two objects that consist of n binary attributes. The comparison of two such objects, i.e., two binary vectors, leads to the following four quantities (frequencies):

f_{00} = the number of attributes where x is 0 and y is 0
f_{01} = the number of attributes where x is 0 and y is 1
f_{10} = the number of attributes where x is 1 and y is 0
f_{11} = the number of attributes where x is 1 and y is 1

6.8 Exploration

Data exploration can aid in selecting appropriate preprocessing and data analysis techniques. It can even address some of the questions typically answered by data mining. For example, patterns can sometimes be found by visually inspecting the data. Also, some of the techniques used in data exploration, such as visualization, can be used to understand and interpret data mining results.

The Iris Data Set

In the following discussion, we will often refer to the Iris data set that is available from the University of California at Irvine (UCI) Machine Learning Repository. It consists of information on 150 Iris flowers, 50 each from one of three Iris species: Setosa, Versicolour, and Virginica. Each flower is characterized by five attributes:
1. Sepal length in centimeters
2. Sepal width in centimeters
3. Petal length in centimeters
4. Petal width in centimeters
5. Class (Setosa, Versicolour, Virginica)

6.8.1 Summary Statistics

Summary statistics are quantities, such as the mean and standard deviation that capture various characteristics of a potentially large set of values with a single number or a small set of numbers. Everyday examples of summary statistics are the average household income or the fraction of college students who complete an undergraduate degree in four years.

Frequencies and the Mode

Given a set of unordered categorical values, there is not much that can be done to further characterize the values except to compute the frequency with which each value occurs for a particular set of data. Given a categorical attribute x, which can take values $\{v_1, \dots, v_i, \dots v_k\}$ and a set of m objects, the frequency of a value v_i is defined as

$$frequency(v_i) = \frac{number\ of\ objects\ with\ attribute\ value\ v_i}{m} \quad (6.2)$$

Class	Size	Frequency
freshman	140	0.33
sophomore	160	0.27
junior	130	0.22
senior	170	0.18

Table 6.8 Class size for students in a hypothetical college

The mode of a categorical attribute is the value that has the highest frequency.

Percentiles

For ordered data, it is more useful to consider the percentiles of a set of values. In particular, given an ordinal or continuous attribute r and a number p between 0 and 100, p between 0 and 100, the p^{th} percentile x_p is a value of x are less than x_p. For instance, the 50^{th} percentile is the value $x_{50\%}$ such that 50% of all values of x are less than $x_{50\%}$. Table 6.9 shows the percentiles for the four quantitative attributes of the Iris data set.

Percentile	Sepal Length	Sepal Width	Petal Length	Petal Width
0	4.3	2.0	1.0	0.1
10	4.8	2.5	1.4	0.2
20	5.0	2.7	1.5	0.2
30	5.2	2.8	1.7	0.4
40	5.6	3.0	3.9	1.2
50	5.8	3.0	4.4	1.3
60	6.1	3.1	4.6	1.5
70	6.3	3.2	5.0	1.8
80	6.6	3.4	5.4	1.9
90	6.9	3.6	5.8	2.2
100	7.9	4.4	6.9	2.5

Table 6.9 Percentiles for sepal length, sepal width, petal length, and petal width. (All values are in centimeters.)

Measures of Location: Mean and Median

For continuous data, two of the most widely used summary statistics are the mean and median, which are measures of the location of a set of values. Consider a set of m objects and an attribute x. Let $\{x_1, \ldots, x_m\}$ be the attribute values of x for these m objects. As a concrete example, these values might be the heights of m children. Let $\{x_{(1)}, \ldots, x_{(m)}\}$ represent the values of x after they have been sorted in non-decreasing order. Thus, $x_{(1)} = \min(x)$ and $x_{(m)} = \max(x)$. Then, the mean and median are defined as follows:

$$mean(x) = \bar{x} = \frac{1}{m}\sum_{i=1}^{m} x_i \qquad (6.3)$$

$$median(x) = \begin{cases} x_{(r+1)} & if\ m\ is\ odd, i.e., m = 2r+1 \\ \frac{1}{2}\left(x_{(r)} + x_{(r+1)}\right) & if\ m\ is\ even, i.e., m = 2r \end{cases} \qquad (6.4)$$

To summarize, the median is the middle value if there are an odd number of values and the average of the two middle values if the number of values is even. Thus, for seven values, the median is $x_{(4)}$, while for ten values, the median is $\frac{1}{2}(x_{(5)} + x_{(6)})$.

Measures of Spread: Range and Variance

Another set of commonly used summary statistics for continuous data are those that measure the dispersion or spread of a set of values. Such measures indicate if the attribute values are widely spread out or if they are relatively concentrated around a single point such as the mean.

The simplest measure of spread is the range, which, given an attribute x with a set of m values $\{x_1, \ldots, x_m\}$, is defined as

$$range(x) = \max(x) - \min(x) = x_{(m)} - x_{(1)} \qquad (6.5)$$

Measure	Sepal Length	Sepal Width	Petal Length	Petal Wi
range	3.6	2.4	5.9	2.4
std	0.8	0.4	1.8	0.8
AAD	0.7	0.3	1.6	0.6
MAD	0.7	0.3	1.2	0.7
IQR	1.3	0.5	3.5	1.5

Table 6.10 Range, standard deviation (std), absolute average difference (AAD), median absolute difference (MAD), and interquartile range (IQR) for sepal length, sepal width, petal length, and petal width. (All values are in centimeters.)

The variance is preferred as a measure of spread. The variance of the (observed) values of an attribute x is typically written as s_x^2 and is defined below. The standard deviation, which is the square root of the variance, is written as s_x and has the same units as x.

$$variance(x) = s_x^2 = \frac{1}{m-1}\sum_{i=1}^{m}(x_i - \bar{x})^2 \qquad (6.6)$$

The mean can be distorted by outliers, and since the variance is computed using the mean, it is also sensitive to outliers. Indeed, the variance is particularly sensitive to outliers since it uses the squared difference between the mean and other values. As a result, more robust estimates of the spread of a set of values are often used. The following are the definitions of three such measures: the absolute average deviation (AAD), the median absolute deviation (MAD), and the interquartile range (IQR). Table 6.10 shows these measures for the Iris data set.

$$AAD(x) = \frac{1}{m}\sum_{i=1}^{m}|x_i - \bar{x}| \qquad (6.7)$$

$$MAD(x) = median(\{|x_1 - \bar{x}|, \ldots, |x_m - \bar{x}|\}) \qquad (6.8)$$

$$interquartile\ range(x) = x_{75\%} - x_{25\%} \qquad (6.9)$$

6.8.2 Visualization

Data visualization is the display of information in a graphic or tabular format. Successful visualization requires that the data (information) be converted into a visual format so that the characteristics of the data and the relationships among data items or attributes can be analyzed or reported. The goal of visualization is the interpretation of the visualized information by a person and the formation of a mental model of the information.

A domain specialist examines visualizations of the data that may be the best way of finding patterns of interest since, by using domain knowledge; a person can often quickly eliminate many uninteresting patterns and direct the focus to the important patterns.

Techniques

Visualization techniques are often specialized to the type of data being analyzed. Indeed, new visualization techniques and approaches, as well as specialized variations of existing approaches, are being continuously created, typically in response to new kinds of data and visualization tasks.

Let us examine the techniques for visualizing data for a small number of attributes. Some of these techniques, such as histograms, give insight into the distribution of the observed values for a single attribute. Other techniques, such as scatter plots, are intended to display the relationships between the values of two attributes.

Stem and Leaf Plots

Stem and leaf plots can be used to provide insight into the distribution of one-dimensional integer or continuous data.

Histograms

Stem and leaf plots are a type of histogram, a plot that displays the distribution of values for attributes by dividing the possible values into bins and showing the number of objects that fall into each bin. For categorical data, each value is a bin.

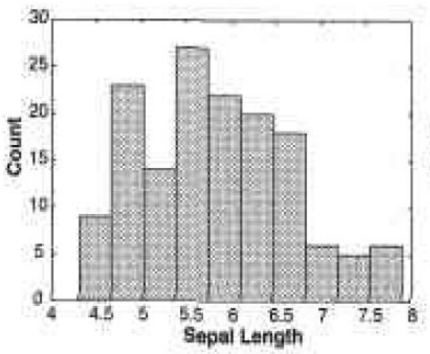


Figure 6.9 Histograms of sepal length

Two-Dimensional Histograms Two-dimensional histograms are also possible. Each attribute is divided into intervals and the two sets of intervals define two-dimensional rectangles of values.

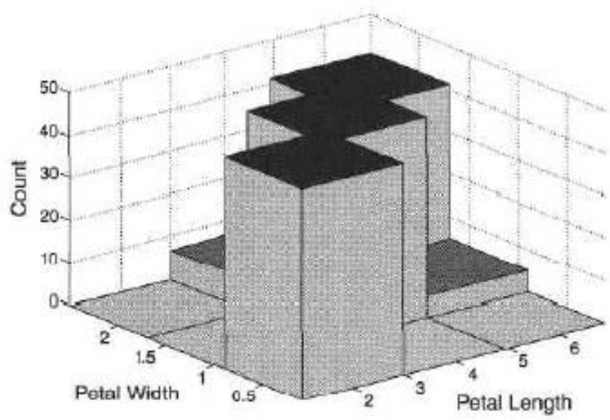


Figure 6.10 Two-dimensional histogram of petal length and width in the iris data set.

Box Plots

Box plots are another method for showing the distribution of the values of a single numerical attribute. Figure 6.11 shows a labeled box plot for sepal length. The lower and upper ends of the box indicate the 25th and 75th percentiles, respectively, while the line inside the box indicates the value of the 50th percentile. The top and bottom lines of the tails indicate the 10th and 90th percentiles. Outliers are shown by "+" mark. Box plots are relatively compact, and thus, many of them can be shown on the same plot. Simplified versions of the box plot, which take less space, can also be used.

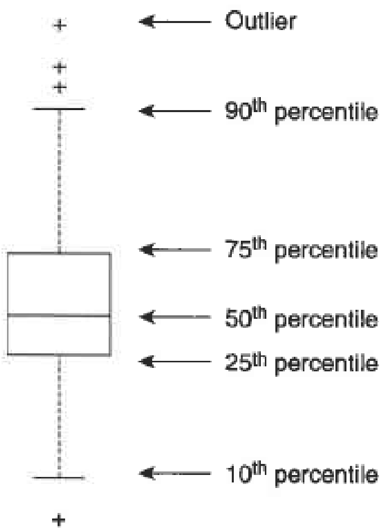


Figure 6.11 Description of box plot for sepal length.

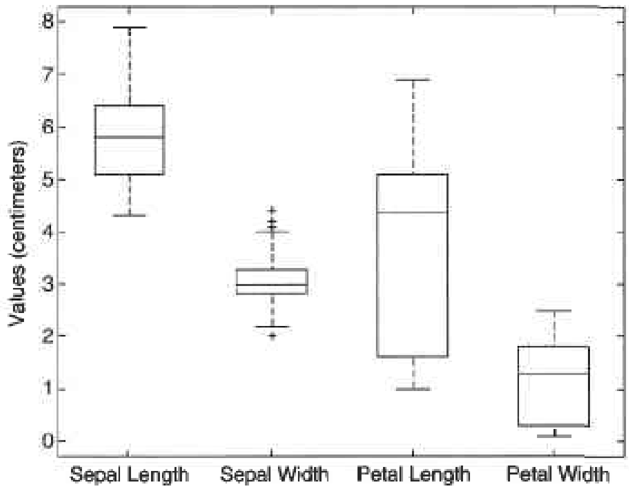


Figure 6.12 Box-plot of Iris dataset

Scatter Plots

Each data object is plotted as a point in the plane using the values of the two attributes as x and y coordinates. It is assumed that the attributes are either integer- or real-valued.

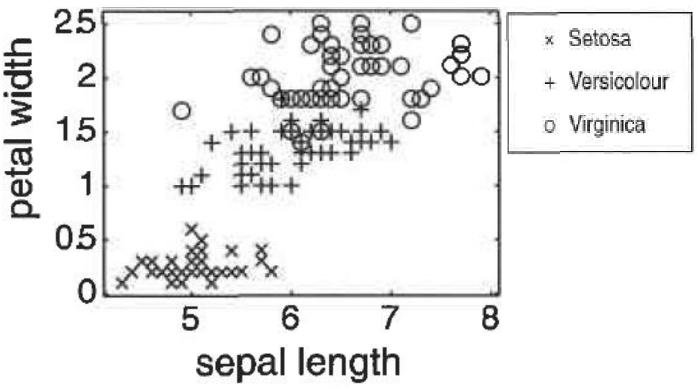


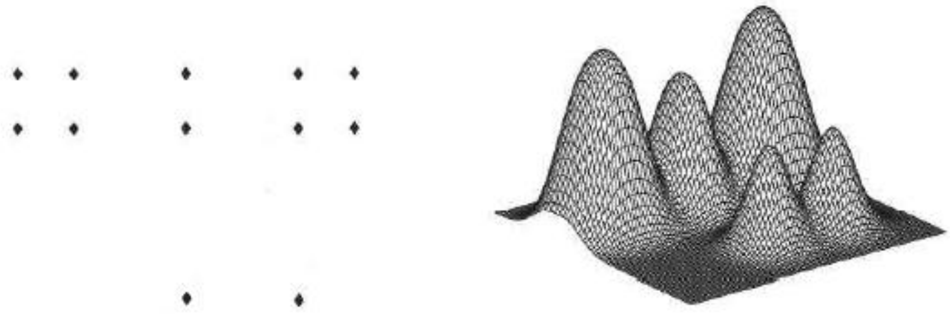
Figure 6.13 Scatter plots for the iris data set

Contour Plots

For some three-dimensional data, two attributes specify a position in a plane, while the third has a continuous value, such as temperature or elevation. A useful visualization for such data is a contour plot, which breaks the plane into separate regions where the values of the third attribute (temperature, elevation) are roughly the same. A common example of a contour plot is a contour map that shows the elevation of land locations.

Surface Plots

Like contour plots, surface plots use two attributes for the x and y coordinates. The third attribute is used to indicate the height above the plane defined by the first two attributes.



(a) Set of 12 points.

(b) Overall density function—surface plot.

Figure 6.14 Density of a set of 12 points

6.9 Check your progress questions

1. Define data object.
2. Define nominal attribute.
3. Define binary attribute.
4. Define ordinal attribute
5. Define numeric attribute.

6.10 Answer to check your progress questions

1. Data sets are made up of data objects. A data object represents an entity. Data objects are described by attributes. Attributes can be nominal, binary, ordinal, or numeric.
2. The values of a nominal (or categorical) attribute are symbols or names of things, where each value represents some kind of category, code, or state.
3. Binary attributes are nominal attributes with only two possible states (such as 1 and 0 or true and false). If the two states are equally important, the attribute is symmetric; otherwise it is asymmetric.
4. An ordinal attribute is an attribute with possible values that have a meaningful order or ranking among them, but the magnitude between successive values is not known.
5. A numeric attribute is quantitative (i.e., it is a measurable quantity) represented in integer or real values. Numeric attribute types can be interval-scaled or ratio-scaled. The values of an interval-scaled attribute are measured in fixed and equal units. Ratio-scaled attributes are numeric attributes with an inherent zero-point.

6.11 Summary

The basic statistical measures for data summarization include mean, weighted mean, median, and mode for measuring the central tendency of data; and range, percentiles, variance, and standard deviation for measuring the dispersion of data. Graphical representations (e.g., boxplots, histograms, and scatter plots) facilitate visual inspection of the data and are thus useful for data preprocessing and mining. Data visualization is the display of information in a graphic or tabular format. Measures of object similarity and dissimilarity are used in data mining applications such as clustering, outlier analysis, and nearest-neighbor classification.

6.12 Keywords

- Dimensionality - It is the number of attributes that the objects in the data set possess.
- Noise - It is the random component of a measurement error.
- Precision – It is the closeness of repeated measurements (of the same quantity) to one another.
- Bias – It is a systematic variation of measurements from the quantity being measured.
- Accuracy – It is the closeness of measurements to the true value of the quantity being measured.
- Aggregation – It is the combining of two or more objects into a single object.
- Sampling - It is a commonly used approach for selecting a subset of the data objects to be analyzed.
- The curse of dimensionality - It refers to the phenomenon that many types of data analysis become significantly harder as the dimensionality of the data increases.
- Feature Creation - It is frequently possible to create, from the original attributes, a new set of attributes that captures the important information in a data set
- Binarization - If there are m categorical values, then uniquely assign each original value to an integer in the interval $[0, m-1]$. Next, convert each of these m integers to a binary number since $n = \lceil \log_2(m) \rceil$.
- Discretization - Transformation of a continuous attribute to a categorical attribute involves two subtasks: deciding how many categories to have and determining how to map the values of the continuous attribute to these categories.

6.13 Self Assessment Questions and Exercises

1. Explain about types of data.
2. Discuss measures of similarity and dissimilarity.
3. Briefly explain about data preprocessing.
4. Give short notes on data exploration.

6.14 Further Reading

1. Poonkuzhali. S, Saravanakumar. C, Data Warehousing & Data Mining, Charulatha Publications.
2. Bhavani Thuraishingham (1999), Data Mining: Technologies, Techniques, Tools, and Trends, CRC Press LLC.
3. Bharat Bhushan Agarwal, Sumit Prakash Tayal, Data Mining and Data Warehousing, University Science Press.
4. Jiawei Han, Micheline Kambar, Jian Pei, Data mining concepts and techniques, Morgan Kaufmann is an imprint of Elsevier.
5. Pang-Ning Tan, Vipin Kumar, Michael Steinbach, Introduction to Data Mining, Pearson.
6. Rao. N. Raghavendra, Global Virtual Enterprises in Cloud Computing Environments, United States of America by IGI Global.

BLOCK - 3 ASSOCIATION RULES

UNIT – 7

INTRODUCTION TO ASSOCIATION RULE ALGORITHM

Structure

7.1 Introduction

Association rule mining is a procedure that is meant to find frequent patterns, correlations, associations, or causal structures from data sets found in various kinds of databases such as relational databases, transactional databases, and other forms of data repositories.

Given a set of transactions, association rule mining aims to find the rules which enable us to predict the occurrence of a specific item based on the occurrences of the other items in the transaction. Association rule mining is the data mining process of finding the rules that may govern associations and causal objects between sets of items.

7.2 Objectives

After going through the unit you will be able to:

- Understand the concept of association rule
- Gain knowledge on association rule algorithm such as A Priori algorithm, Partition Algorithm, Pincer-Search Algorithm, etc.

7.3 Methods to Discover Association Rules

For a given transaction database T, an ***association rule*** is an expression of the form $X \Rightarrow Y$, where X and Y are subsets of A and $X \Rightarrow Y$ holds with ***confidence*** τ, if τ% of transactions in D that support X also support Y. The rule $X \Rightarrow Y$ has ***support*** σ in the transaction set T if σ% of transactions in T support $X \cup Y$.

The intuitive meaning of such a rule is that a transactions of the database which contains X tends to contain Y. Given a set of transactions, T, the problem of mining association rules is to discover all rules that have support and confidence greater than or equal to the user-specified minimum support and minimum confidence, respectively.

7.3.1 Problem Decomposition

The problem of mining association rules can be decomposed into two subproblems:

Find all sets of items (itemsets) whose support is greater than the user-

specified minimum support, σ. Such itemsets are called frequent itemsets. Use the frequent itemsets to generate the desired rules.

The general idea is that if, say $ABCD$ and AB are frequent itemsets,

then we can determine if the rule $AB \Rightarrow CD$ holds by checking the following inequality

$$\frac{s(\{A,B,C,D\})}{s(\{A,B\})} \geq \tau,$$

Where $s(X)$ is the support of X in T.

Frequent Set

Let T be the transaction database and σ be the user-specified minimum support. An itemset $X \subseteq A$ is said to be a frequent itemset

in T with respect to σ, if $s(X)_r \geq \sigma$

For example, if we assume σ=50%, then $\{bread, jam, butter\}$ is a

frequent set as it is supported by at least 3 out of 6 transactions. We

can see that any subset of this set is also a frequent set.

Maximal Frequent Set

A frequent set is a maximal frequent set if it is a frequent set and no superset of this is a frequent set.

Border Set

An itemset is a border set if it is not a frequent set, but all its proper

subsets are frequent sets.

Example 7.1

Study the following transaction database. We shall use this database for illustration of following algorithm.

$A = \{A1, A2, A3, A4, A5, A6, A7, A8, A9\}$. Assume σ=20%. Since T contains 15 records, it means that an itemset that is supported by at least three transactions is a frequent set.

A1	A2	A3	A4	A5	A6	A7	A8	A9
1	0	0	0	1	1	0	1	0
0	1	0	1	0	0	0	1	0
0	0	0	1	1	0	1	0	0
0	1	1	0	0	0	0	0	0
0	0	0	0	1	1	1	0	0
0	1	1	1	0	0	0	0	0
0	1	0	0	0	1	1	0	1
0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	1	0
0	0	1	0	1	0	1	0	0
0	0	1	0	1	0	1	0	0
0	0	0	0	1	1	0	1	0
0	1	0	1	0	1	1	0	0
1	0	1	0	1	0	1	0	0
0	1	1	0	0	0	0	0	1

Table 7.1 Sample Database

X	SUPPORT COUNT
{1}	2
{2}	6
{3}	6
{4}	4
{5}	8
{6}	5
{7}	7
{8}	4
{9}	2
{5, 6}	3
{5, 7}	5
{6, 7}	3
{5, 6, 7}	1

Table 7.2 Frequent count for Some Itemsets

The numbers of transactions supporting some of the itemsets are given in Table 3.2. {5, 6, 7} is a border set; {5, 6} is a maximal frequent set; {2, 4} is also a maximal frequent set. But there is no border set having {2, 4} as a

proper subset. Thus, {2, 4} and {5, 6, 7} jointly represent the set of

all frequent sets of T with respect to σ. This is so, because we can generate all the frequent sets from these two itemsets.

7.4 A Priori Algorithm

It is also called the level-wise algorithm. It was proposed by

Agrawal and Srikant in 1994. It is the most popular algorithm to

find all the frequent sets. The first pass of the algorithm counts item occurrences to determine the frequent itemsets. A subsequent pass,

say pass k, consists of two phases. First, the frequent itemsets L_{k-1}

found in the $(k-1)^{th}$ pass are used to generate the candidate itemsets C_k, using the a priori candidate generation procedure described below.

Next, the database is scanned and the support of candidates in C_k is counted. The set of candidate itemsets is subjected to a pruning

process to ensure that all the subsets of the candidate sets are

already known to be frequent itemsets. The candidate generation

process and the pruning process are the most important parts of this algorithm.

7.4.1 Candidate Generation

Given L_{k-1}, the set of all frequent $(k-1)$-itemsets. Let us assume

that the set of frequent 3-itemsets are {1, 2, 3}, {1, 2, 5}, {1, 3, 5},

{2, 3, 5}, {2, 3, 4}. Then the 4-itemsets that are generated as candidate itemsets are the supersets of these 3-itemsets and in addition, all the 3-itemset subsets of any candidate 4-itemset must

be already known to be inL_3.

The candidate –generation method is described below.

gen_candidate_itemsets with the given L_{k-1} as follows:

$C_k = \emptyset$
for all itemsets $l_1 \in L_{k-1}$ ***do***
for all itemsets $l_2 \in L_{k-1}$ ***do***
  if $l_1[1] = l_2[1] \wedge l_1[2] = l_2[2] \wedge \ldots \wedge l_1[k-1] < l_2[k-1]$
  then $c = l_1[1], l_1[2] \ldots l_1[k-1], l_2[k-1]$
  $C_k = C_k \cup \{c\}$

7.4.2 Pruning

The pruning step eliminates the extensions of $(k-1)$-itemsets which are not found to be frequent, from being considered for counting support. For example, for C_4, the itemset {2,3,4,5} is pruned, since all its 3-subsets are not in L_3.

The pruning algorithm is described below.

prune(C_k)

for all $c \in C_k$
for all $(k-1)$-subsets d of c ***do***
  if $d \notin L_{k-1}$
  then $C_k = C_k \setminus \{c\}$

A priori Algorithm is described below

A Priori Algorithm

Initialize: $k := 1$, C_1 = all the 1-itemsets;
read the database to count the support of C_1 to determine L_1.
$L_1 :=$ {frequent 1-itemsets};
$k := 2$; // *k represents the pass number*//
while $(L_{k-1} \neq \emptyset)$ ***do***
begin
  $C_k :=$ ***gen_candidate_itemsets with the given*** L_{k-1}
  prune(C_k)
  for all transactions $t \in T$ ***do***
  increment the count of all candidates in C_k that are contained in t;
  $L_k :=$ All candidates in C_k with minimum support ;
  $k := k + 1$;
end
Answer $:= \cup_k L_k$;

7.4.3 A Priori Algorithm By Example

We illustrate the working of the algorithm with Example 7.1 discussed above.

k:=1

Read the database to count the support of 1-itemsets (Table 7.3).

The frequent 1-itemsets and their support counts are given below.

{1}	2
{2}	6
{3}	6
{4}	4
{5}	8
{6}	5
{7}	7
{8}	4
{9}	2

Table 7.3 The frequent 1-itemsets and their support counts

L_1 is Collecting those items that satisfy the minimum support (here minimum support is greater than 2) .

L_1:= { {2}→6. {3}→6, {4}→4, {5}→8, {6}→5, {7}→7, {8}→4 }

k:=2

In the candidate generation step, we get

C_2 := { {2,3}, {2,4}, {2, 5}, {2,6}, {2, 7}, {2, 8}, {3, 4}, {3, 5}, {3, 6}, {3,
{4, 5}, {4, 6}, {4,7}, {4, 8},{5, 6}, {5, 7}, {5, 8}, {6,7}, {6, 8}, {7,

The pruning step does not change C_2.

Read the database to count the support of elements in C_2 to get

L_2 := { {2, 3}→3, {2,4}→3, {3, 5}→3, {3, 7}→3, {5, 6}→3, {5, 7}→5,

k:=3

In the candidate generation step, using {2,3} and {2,4}, we get {2,3,4}using {3,5} and {3,7}, we get {3,5,7} and similarly from {5,6} and {5,7}, we get {5,6,7}.

$C_3 := \{\{2, 3, 4\}, \{3, 5, 7\}, \{5, 6, 7\}\}.$

The pruning step prunes {2, 3, 4} as not all subsets of size 2, i.e., {2,3},{2,4},{3,4} are presented in L_3. The other two itemsets are retained. Thus the pruned C_3 is {{3, 5, 7} , {5, 6, 7}}.

Read the database to count the support of the itemsets in C_3 to get $L_3 := \{\{3, 5, 7\} \rightarrow 3\}$.

k:=4

Since L_3 contains only one element, C_4 is empty and hence the algorithm stops, returning the set of frequent sets along with their respective support values as $L := L_1 \cup L_2 \cup L_3$

7.5 Partition Algorithm

The partition algorithm uses two scans of the database to discover all frequent sets. In one scan, it generates a set of all frequent itemsets by scanning the database once. This is a superset of all frequent itemsets, i.e., it may contain false positives. During the second scan, counters for each of these itemsets are set up and their actual support is measured in one scan of the database.

The algorithm executes in two phases. In the first phase, the partition algorithm divides the database into a number of non-overlapping partitions. The partitions are considered one at a time and all frequent itemsets for that partition are generated. Thus, if there are *n* partitions, Phase I of the algorithm takes *n* iterations.

At the end of Phase I, these frequent itemsets are merged to generate a set of all potential frequent itemsets. In this step, the local frequent itemsets of same lengths from all *n* partitions are combined to generate the global candidate itemsets.

In Phase II, the actual supports for these itemsets are generated and the frequent itemsets are identified. The algorithm reads the entire database once during Phase I and once during Phase II.

The partition sizes are chosen such that each partition can be accommodated in the main memory, so that the partitions are read only once in each phase.

Partition Algorithm

P = partition_database(T); n = Number of partitions
// Phase I
 for i = 1 to n ***do begin***
  read_in_partition(T_i in P)
  L^i = generate all frequent itemsets of T_i using *a priori* method in main n
 end

// Merge Phase
 for ($k = 2$; $L_k^i \neq \emptyset$, $i = 1, 2, \ldots, n$; k++) ***do begin***
  $C_k^G = \bigcup_{i=1}^{n} L_i^k$
 end

// Phase II
 for i = 1 to n ***do begin***
  read_in_partition(T_i in P)
  for all candidates $c \in C^G$ compute $s(c)_{T_i}$
 end
 $L^G = \{c \in C^G \mid s(c)_{T_i} \geq \sigma\}$
 Answer = L^G

Let us take same database T, given in Example 7.1, and the same,

σ. Let us partition, T into three partitions T_1, T_2, and T_3, each containing 5 transactions. The first partition T_1 contains transactions

1 to 5, T_2 contains transactions 6 to 10, and similarly, T_3 contains transactions 11 to 15. Let us fix the local support as equal to the

given support i.e., 20%. Thus $\sigma_1 = \sigma_2 = \sigma_3 = \sigma =$ 20%. Any

item set that appears in just one of the transactions in any partition

is a local frequent set in that partition.

The local frequent sets of the T_1 partition are the itemsets X, such

that $s(X)_{T1} \geq \sigma_1$.

L^1 := { {1}, {2}, {3}, {4}, {5}, {6}, {7}, {8}, {1, 5}, {1, 6}, {1, 8}, {2,
{2, 8}, {4, 5}, {4, 7}, {4, 8}, {5, 6}, {5, 8}, {5, 7}, {6, 7}, {6, 8
{1,5,6}, {1,5,8}, {2,4,8}, {4,5,7}, {5,6,8}, {5,6,7}, {1,5,6,8} }

Similarly,

L^2 := { {2}, {3}, {4}, {5}, {6}, {7}, {8}, {9}, {2,3}, {2,4}, {2,6}, {2,7}, {2,9}, {3,4}, {3, 5}, {3, 7}, {5, 7}, {6,7}, {6,9}, {7,9}, {2,3,4}, {2,6,7}, {2,6,9}, {2,7,9}, {3, 5, 7}, {2,6,7,9} }

L^3 := { {1}, {2}, {3}, {4}, {5}, {6}, {7}, {8}, {9}, {1,3}, {1,5}, {1,7}, {2,3}, {2,4}, {2,6}, {2,7}, {2,9}, {3,5}, {3,7},{3,9},{4,6}, {4,7}, {5,6}, {5,7}, {5,8}, {6,7}, {6,8}, {1,3,5}, {1,3,7}, {1,5,7}, {2,3,9}, {2,4,6}, {2,4,7}, {3,5,7}, {4,6,7}, {5,6,8}, {1, 3, 5, 7}, {2, 4, 6, 7} }

In Phase II, we have the candidate set as

$C := L^1 \cup L^2 \cup L^3$

C := { {1}, {2}, {3}, {4}, {5}, {6}, {7}, {8}, {9}, {1,3}, {1, 5}, {1, 6}, {1,7}, {1, 8}, {2,3}, {2, 4}, {2,6}, {2,7}, {2, 8}, {2,9},{3,4}, {3,5}, {3,7}, {3,9}, {4, 5}, {4,6}, {4,7}, {4, 8}, {5, 6},{5,7}, {5, 8}, {5, 7}, {6, 7}, {6, 8}, {6,9}, {7,9},{1,3,5}, {1,3,7}, {1,5,6}, {1,5,7}, {1,5,8}, {1,6,8}, {2,3,4}, {2,3,9}, {2,4,6}, {2,4,7}, {2,4,8}, {2,6,7}, {2,6,9}, {2,7,9}, {3,5,7}, {4,5,7}, {4,6,7}, {5,6,8}, {5,6,7}, {1,5,6,8}, {2,6,7,9}, {1, 3, 5, 7}, {2, 4, 6, 7} }

Read the database once to compute the global support of the sets in C and get the final set of frequent sets.

7.6 Pincer-Search Algorithm

A priori algorithm operates in a bottom-up, breadth-first search method. The computation starts from the smallest set of frequent itemsets and moves upward till it reaches the largest frequent itemset. The number of database passes is equal to the largest size of the frequent itemset. When any one of the frequent itemsets becomes longer, the algorithm has to go through many iterations and, as a result, the performance decreases. A natural way to overcome this difficulty is to somehow incorporate a bi-directional search, which takes advantage of both the bottom-up as well as the top-down process.

The pincer-search algorithm is based on this principle. It attempts to find the frequent itemsets in a bottom-up manner but, at the same time, it maintains a list of maximal frequent itemsets. While making a database pass, it also counts the support of these candidate maximal frequent itemsets to see if anyone of these is frequent. In that event, it can conclude that all the subsets of these frequent sets are going to be frequent and, hence, they are not verified for the support count in the next pass. There is a chance to discover a very large maximal frequent itemset very early in the algorithm.

If this set subsumes all the candidate sets of level, then need not precede further and thus save many database passes. Pincer-search has an advantage over a priori algorithm when the largest frequent itemset is long. In this algorithm, in addition to counting the supports of the candidate in the bottom-up direction, it also counts the supports of the itemsets of some itemsets using

a top-down approach. These are called the Maximal Frequent Candidate Set (MFCS).

Pincer-Search Method

```
L_0 := ∅; k := 1; C_1 := {{i} | i ∈ I}; S_0 = ∅;
MFCS := {{1,2, …, n}}; MFS := ∅;
do until C_k = ∅ and S_{k-1} = ∅
        read database and count supports for C_k and MFCS;
        MFS := MFS ∪ {frequent itemsets in MFCS};
        S_k := {infrequent itemsets in C_k};
        call MFCS-gen algorithm if S_k ≠ ∅;
        call MFS-pruning procedure;
        generate candidates C_{k+1} from C_k; (similar to a priori's generate & prune)
        if any frequent itemset in C_k is removed in MFS-pruning procedure
            call the recovery procedure to recover candidates to C_{k+1};
        call MFCS prune procedure to prune candidates in C_{k+1};
        k := k+1;
return MFS
```

MFCS-gen

```
for all itemsets s ∈ S_k
        for all itemsets m ∈ MFCS
         if s is a subset of m
         MFCS := MFCS\{m};
         for all items e ∈ itemset s
           if m\{e} is not a subset of any itemset in MFCS
                   MFCS := MFCS ∪ {m\{e}};
return MFCS
```

Recovery

```
for all itemsets l ∈ C_k
        for all itemsets m ∈ MFS
         if the first k–1 items in l are also in m
                 /* suppose m.item_j = l.item_{k-1} */
                 for i from j+1 to |m|
                  C_{k+1} := C_{k+1} ∪ {{l.item_1, l.item_2, …, l.item_k, m.item_i}}
```

MFS-Prune

```
        for all itemsets c in C_k
        if c is a subset of any itemset is the current MFS
        delete c from C_k;
```

MFCS-Prune

```
        for all itemsets c in C_{k+1}
        if c is not a subset of any itemset in the current MFCS
        delete c from C_{k+1};
```

Let us use Example 7.1 to illustrate the working of the sample.

STEP 1: $L_0 := \varnothing; k := 1;$

$C_1 := \{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}, \{6\}, \{7\}, \{8\}, \{9\}\}$

MFCS := {1, 2, 3, 4, 5, 6, 7, 8, 9}

MFS := ∅;

PASS ONE: Database is read to count the support as follows

{1} → 2, {2} → 6, {3} → 6, {4} → 4, {5} → 8, {6} → 5, {7} → 7, {8} → 4, {9}→2

{1, 2, 3, 4, 5, 6, 7, 8, 9} → 0.

So MFCS := {1, 2, 3, 4, 5, 6, 7, 8, 9} and MFS := ∅;

$L_1 := \{\{2\}, \{3\}, \{4\}, \{5\}, \{6\}, \{7\}, \{8\}\}$

$S_1 := \{\{1\}, \{9\}\}$

At this stage call the MFCS-gen to update MFCS.

For {1} in S_1 and for {1, 2, 3, 4, 5, 6, 7, 8, 9} in MFCS, we get the new element in MFCS as {2, 3, 4, 5, 6, 7, 8}.

For {9} in S_1 and for {2, 3, 4, 5, 6, 7, 8, 9} in MFCS, we get the new element in MFCS as {2, 3, 4, 5, 6, 7, 8}.

We generate the candidate itemsets

C_2 := { {2,3}, {2,4}, {2,5}, {2,6}, {2,7}, {2,8}, {3,4}, {3,5}, {3,6}, {3,7}, {3,8}, {4,5}, {4,6}, {4,7}, {4,8}, {5,6}, {5,7}, {5,8}, {6,7}, {6,8}, {7,8} }

PASS TWO: read the database to count the support of elements in C_2 and MFCS as given below:

{2,3}→3, {2,4}→3, {2,5}→0, {2,6}→2, {2,7}→2, {2,8}→1, {3,4}→1, {3,5}→3, {3,6}→0, {3,7}→3, {3,8}→0, {4,5}→1, {4,6}→1, {4,7}→2, {4,8}→1, {5,6}→3, {5,7}→5, {5,8}→2, {6,7}→3, {6,8}→2, {7,8}→0

{2, 3, 4, 5, 6, 7, 8}→0.

MFS:= ∅;

L_2 := { {2,3}, {2,4}, {3,5}, {3,7}, {5,6}, {5,7}, {6,7} }

S_2 := { {2,5}, {2,6}, {2, 7} {2,8}, {3,4}, {3,6}, {3,8}, {4,5}, {4,6}, {4,7}, {4,8}, {5,8}, {6,8}, {7,8} }

For {2, 5} in S_2 and for {2, 3, 4, 5, 6, 7, 8} in MFCS, we get the new element in MFCS as { 3, 4, 5, 6, 7, 8} and {2, 3, 4, 6, 7, 8}.

For {2, 6} in S_2 and for { 3, 4, 5, 6, 7, 8} in MFCS, since {2,6} is not

contained in this element of MFCS and hence, no action.

For {2, 3, 4, 6, 7, 8} we get two new elements in MFCS in place of {2, 3, 4, 6, 7, 8} as {3, 4, 6, 7, 8} and {2, 3, 4, 7, 8}. Since {3, 4, 6, 7, 8} is already contained in an element of MFCS, it is excluded from MFCS.So at this stage MFCS:={{3, 4, 5, 6, 7, 8}, {2, 3, 4, 7, 8}}

For {2,7} in S_2, we get

MFCS := {{3, 4, 5, 6, 7, 8}, {2, 3, 4, 8}}.

For {2,8} in S_2, we get

MFCS := {{3, 4, 5, 6, 7, 8}, {2, 3, 4}}.

For {3,4} in S_2, we get

MFCS := {{3, 5, 6, 7, 8}, {4, 5, 6, 7, 8}, {2, 3}, {2, 4}}.

For {3,6} in S_2, we get

MFCS := {{3, 5, 7, 8}, {4, 5, 6, 7, 8}, {2, 3}, {2, 4}}.

For {3,8} in S_2, we get

MFCS := {{3, 5, 7}, {4, 5, 6, 7, 8}, {2, 3}, {2, 4}}.

For {4,5} in S_2, we get

MFCS := {{3, 5, 7}, {5, 6, 7, 8}, {4, 6, 7, 8}, {2, 3}, {2, 4}}.

For {4,6} in S_2, we get

MFCS := {{3, 5, 7}, {5, 6, 7, 8}, {4, 7, 8}, {2, 3}, {2, 4}}.

For {4,7} in S_2, we get

MFCS := {{3, 5, 7}, {5, 6, 7, 8}, {4, 8}, {2, 3}, {2, 4}}.

For {4,8} in S_2, we get

MFCS := {{3, 5, 7}, {5, 6, 7, 8}, {2, 3}, {2, 4}}.

For {5,8} in S_2, we get

MFCS := {{3, 5, 7}, {6, 7, 8}, {5, 6, 7}, {2, 3}, {2, 4}}.

For {6, 8} in S_2, we get

MFCS := {{7, 8}, {3, 5, 7}, {5, 6, 7}, {2, 3}, {2, 4}}.

For {7, 8} in S_2, we get

MFCS := {{8}, {3, 5, 7}, {5, 6, 7}, {2, 3}, {2, 4}}.

We generate the candidate sets as

C_3 := {{2, 3, 4}, {3, 5, 7}, {5, 6, 7}}

In the pruning stage the itemsets{2, 3, 4} are pruned from C_3 and hence,

C_3 := {{3, 5, 7}, {5, 6, 7}}

At this stage we make one more pass of the database to count the supports

{{3, 5, 7}, {5, 6, 7}}.

7.7 Check your progress questions

1. Define association rule mining.
2. Define maximal frequent set.
3. Define border set.

7.8 Answer to check your progress questions

1. Association rule mining consists of first finding frequent itemsets (sets of items, such as *A* and *B*, satisfying a *minimum support threshold*, or percentage of the task relevant tuples), from which strong association rules in the form of $A \Rightarrow B$ are generated.

2. A frequent set is a maximal frequent set if it is a frequent set and no superset of this is a frequent set.
3. An itemset is a border set if it is not a frequent set, but all its proper subsets are frequent sets.

7.9 Summary

The discovery of frequent patterns, associations, and correlation relationships among huge amounts of data is useful in selective marketing, decision analysis, and business management. A popular area of application is market basket analysis, which studies customers' buying habits by searching for itemsets that are frequently purchased together (or in sequence). Association rule mining consists of first finding frequent itemsets, from which strong association rules in the form of $A \Rightarrow B$ are generated. The Apriori algorithm is a seminal algorithm for mining frequent itemsets for Boolean association rules.

7.10 Keywords

- **A priori Algorithm** - Apriori is an algorithm for frequent item set mining and association rule learning over relational databases. It proceeds by identifying the frequent individual items in the database and extending them to larger and larger item sets as long as those item sets appear sufficiently often in the database.
- **Partition Algorithm** - The partition algorithm uses two scans of the database to discover all frequent sets. In one scan, it generates a set of all frequent itemsets by scanning the database once. This is a superset of all frequent itemsets, i.e., it may contain false positives. During the second scan, counters for each of these itemsets are set up and their actual support is measured in one scan of the database.
- **Pincer-Search Algorithm** - The pincer-search algorithm is based on this principle. It attempts to find the frequent itemsets in a bottom-up manner but, at the same time, it maintains a list of maximal frequent itemsets.

7.11 Self Assessment Questions and Exercises

1. Explain in detail about A Priori algorithm
2. Briefly discuss the advantages of Pincer –Search Algorithm over A Priori and partition algorithm.
3. Write a note on Partition algorithm.

7.12 Further Reading

1. Arun K Pujari, Data Mining Techniques, Universities Press
2. Poonkuzhali. S, Saravanakumar. C, Data Warehousing & Data Mining, Charulatha Publications.
3. Jiawei Han, Micheline Kambar, Jian Pei, Data mining concepts and techniques, Morgan Kaufmann is an imprint of Elsevier.
4. Bhavani Thuraishingham (1999), Data Mining: Technologies, Techniques, Tools, and Trends, CRC Press LLC.
5. Bharat Bhushan Agarwal, Sumit Prakash Tayal, Data Mining and Data Warehousing, University Science Press.
6. Pang-Ning Tan, Vipin Kumar, Michael Steinbach, Introduction to Data Mining, Pearson.
7. Rao. N. Raghavendra, Global Virtual Enterprises in Cloud Computing Environments, United States of America by IGI Global.

UNIT – 8

DYNAMIC ITEMSET AND FP TREE GROWTH ALGORITHM

Structure

8.1 Introduction

Dynamic Itemset Counting (DIC) algorithm was proposed by Bin *et al*. in 1997. DIC works like a train running over the data, with stops at intervals *M* between transactions. When the train reaches the end of the transaction file, it has made one pass over the data, and it starts all over again from the beginning for the next pass.A new class of algorithms has been proposed which avoids the generation of large numbers of candidate sets. This method was proposed by Han *et al*. The main idea of the algorithm is to maintain a Frequent Pattern Tree (FP – Tree) of the database. One such method is called the FP-Tree Growth algorithm.

8.2 Objectives

After going through the unit you will be able to understand the concept of dynamic itemset counting algorithm and FP tree growth algorithm.

8.3 Dynamic Item set Algorithm

In A priori algorithm, all itemsets get on the start of a pass and get off at the end. The 1-itemsets take the first pass, the 2-itemsets take the second pass, and so on. In DIC, there is a flexibility of allowing itemsets to get on at any stop as long as they get off at the same stop the next time the train goes around. Therefore, the itemset has seen all the transactions in the file. Initially identify certain ‘stops’ in the database.

Here four different structures are defined:

- Dashed Box
- Dashed Circle
- Solid Box
- Solid Circle

During the execution of the algorithm, at any stop point, the following events take place.

- Certain itemsets in the dashed circle move into the dashed box. These are the itemsets whose support-counts reach σ value during this iteration (reading records between two consecutive stops).
- Certain itemsets enter afresh into the system and get into the dashed circle. These are essentially the supersets of the itemsets that move from the dashed circle to the dashed box.
- The itemsets that have completed one full pass, move from the dashed structure (dashed circle or dashed box) to solid structure (solid circle or solid box).

Formal Description of DIC Algorithm

DIC Algorithm

```
Initially,
    Solid box contains the empty itemset;
    Solid circle is empty
    Dashed box is empty;
    Dashed circle contains all 1-itemsets with the respective stop-number as 0;
    Current stop-number := 0;

do until the dashed circle is empty

read the database till the next stop point and increase the counters of the itemsets in the dashed box
and in the dashed circle as we go along, record by record, to reach the next stop.

increase the current-stop-number by 1;
for each itemset in the dashed circle
   if count of the itemset is greater than σ
   then move the itemset to the dashed box
        generate a new itemset to be put into the dashed circle
        with counter value = 0 and stop number = current stop number.
   else
        if its stop number is equal to the current stop number
        then move this itemset to solid circle.

for each itemset in the dashed box
   if  its stop-number is equal to the current stop number
   then move this itemset to the solid box
end
return the itemsets in solidbox
```

Example 8.1

The following example (Table 7.1) illustrates the working of the DIC algorithm.

Table 7.1

A1	A2	A3	A4	A5	A6	A7	A8	A9
1	0	0	0	1	1	0	1	0
0	1	0	1	0	0	0	1	0
0	0	0	1	1	0	1	0	0
0	1	1	0	0	0	0	0	0
0	0	0	0	1	1	1	0	0
0	1	1	1	0	0	0	0	0
0	1	0	0	0	1	1	0	1
0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	1	0
0	0	1	0	1	0	1	0	0
0	0	1	0	1	0	1	0	0
0	0	0	0	1	1	0	1	0
0	1	0	1	0	1	1	0	0
1	0	1	0	1	0	1	0	0
0	1	1	0	0	0	0	0	1

Step 1: The empty itemset is in the solid box. All the 1-itemsets are in the dashed circles. The sets {1}, {2}, {3}, {4}, {5}, {6}, {7}, {8}, and {9} are in the dashed circle.

Step 2: First read 5 transactions ($t_1 - t_5$). For each transaction, increment the counts of the respective counters of the itemsets in the dashed box or dashed circle. The counts of the itemsets in the dashed circles are given below.{1}=1, {2}=1, {3}=1, {4}=2, {5}=3, {6}=2, {7}=2, {8}=2, and {9}=0. Thus, itemset {5} is put in a dashed box and removed from the dashed circle.

Step 3: Read the next 5 transactions ($t_6 - t_{10}$). The updated counts of the itemsets in the dashed box and in the dashed circle are .{1}=1, {2}=3, {3}=3, {4}=3, {5}=5, {6}=3, {7}=4, {8}=3, and {9}=1. The itemsets {2}, {3}, {4}, {5}, {6}, {7}, and {8}are now put in dashed box and are removed from the dashed circle. The additional itemsets which are put in the dashed circle are

{2, 3}, {2, 4}, {2, 5 }, {2, 6}, {2, 7}, {2, 8}, {3, 4}, {3, 5}, {3, 6}, {3, 7}, {3, 8}, {4, 5}, {4, 6}, {4, 7}, {4, 8}, {5, 6}, {5, 7}, {5, 8}, {6, 7}, {6, 8}, {7, 8}.

Counter for each of these itemsets are included.

Step 4: Read the next 5 transactions ($t_{11} - t_{15}$). The updated count of each of the counters are

{1} = 2, {2} = 5, {3} = 6, {4} = 4, {5} = 8, {6} = 5, {7} = 7, {8} = 4,
{9} = 2. {2, 3} = 1, {2, 4} = 1, {2, 5}= 0, {2, 6} = 1, {2, 7}= 1, {2, 8} =
{3, 4} = 0, {3, 5} = 2, {3, 6} = 0, {3, 7}= 2, {3, 8} = 0, {4, 5}= 0, {4, 6}=
{4, 7}= 1, {4, 8}= 0, {5, 6}= 1, {5, 7}= 2, {5, 8}= 1, {6, 7}= 1, {6, 8}=
{7, 8} = 0.

The 1-itemsets have completed their traversal of the whole set of transactions and hence, {2}, {3}, {4}, {5}, {6}, {7}, and {8} are put in the solid box and removed from the dashed box. The itemset {1} and {9} are removed from the dashed circle and is put in the solid circle. The counters for these sets are now dropped.

Step 5: Read the first 5 transactions ($t_1 - t_5$) and update the count of the itemsets in the dashed box and the dashed circle. The updated counts are

{2, 3} = 1, {2, 4} = 2, {2, 5}= 0, {2, 6} = 1, {2, 7}= 1, {2, 8} = 1, {3,
{3, 5} = 2, {3, 6} = 0, {3, 7} = 2, {3, 8} = 0, {4, 5}= 1, {4, 6}= 1,
{4, 7}= 2, {4, 8}= 1, {5, 6}= 3, {5, 7}= 4, {5, 8}= 2, {6, 7}= 2, {6,
{7, 8} = 0.

At this stage, {5, 6} and {5, 7} are moved from the dashed circle to the dashed box, as their count exceeds the required support.

Step 6: Read the transactions ($t_6 - t_{10}$). The updated counts are

{2, 3} = 2, {2, 4} = 3, {2, 5}= 0, {2, 6} = 2, {2, 7}= 2, {2, 8} = 1, {3,
{3, 5} = 3, {3, 6} = 0, {3, 7}= 3, {3, 8} = 0, {4, 5}= 0, {4, 6}= 1,
{4, 7}= 2, {4, 8}= 1, {5, 6}= 3, {5, 7}= 5, {5, 8}= 2, {6, 7}= 3, {6,
{7, 8} = 0.

Clearly, the itemsets {2, 4}, {3, 5}, {3, 7} and {6, 7} are moved to the solid box from the dashed circle; and {5, 6} and {5, 7} are

moved from the dashed box to the solid box. The remaining itemsets (listed below) in the dashed circle are moved to the solid circle.

{2, 3}, {2, 5}, {2, 6}, {2, 7}, {2, 8}, {3, 4}, {3, 5}, {3, 6}, {3, 7}, {3, 8}, {4, 5}, {4, 6}, {4, 7}, {4, 8}, {5, 8}, {6, 8}, {7, 8}.

The counters for these sets are dropped. The itemset {5, 6, 7} is now put in the dashed circle and a counter for this is introduced.

Step 7: Now scan the transactions $(t_{11} - t_{15})$ and the transactions $(t_1 - t_5)$ to count the support of {5, 6, 7}. Thus, the algorithm requires only 2.75 database passes instead of 3 passes as in the level-wise method.

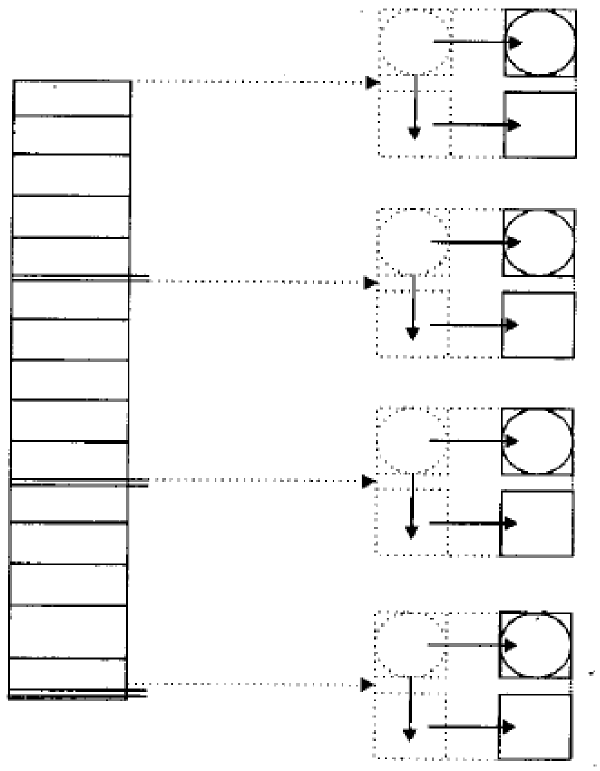


Figure 8.1 A pictorial depiction of the working of DIC. The horizontal arrow indicates movement of the itemset after completion of database pass. The vertical arrow indicates movement of the itemset after reaching the frequency threshold. The dashed arrow indicates the new itemset entering the system at a stop point.

8.4 FP-Tree Growth Algorithm

Most of the algorithms such as A priori, Partition, Pincher search and DIC algorithms suffer from the following two shortcomings:

1. It is costly to handle large numbers of candidate sets.
2. It is tedious to repeatedly scan the database and check a large set of candidates by pattern matching.

The FP-Tree Growth algorithm involves two phases. In Phase I, it constructs the FP-tree with respect to a givenσ. The construction of this tree requires two passes over the whole database. In Phase II, the algorithm does not use the transaction database anymore, but it uses the FP-tree.

Definition

A frequent pattern tree(FP-tree) is a tree structure consisting of an item-prefix-tree and a frequent-item-header table.

Item-prefix-tree:

- It consists of a root node labeled null
- Each non-root node consists of three fields: Item name,
- support count, and node link.

Frequent-item-header-table:

- Item name;
- Head of node link which points to the first node in the FP-
- Tree carrying the item name.

FP-Tree is dependent on the support thresholdσ. For different values of σ, the trees are different. It also depends on the ordering of the items. The ordering that is followed is the decreasing order of

the support counts. However, different ordering may offer differ advantages. Thus, the header table is arranged in this order of the frequent items.

FP-tree construction Algorithm

```
create a root node root of the FP-Tree and label it as null.
    do for every transaction t
        if t is not empty
        insert (t, root)
        link the new nodes to other nodes with similar labels links originating from header list.
    end do
return FP-Tree

insert(t, any_node)
    do while t is not empty
        if any_node has a child node with label head_t
        then increment the link count between any_node and head_t by 1
        else create a new child node of any_node with labe; head_t with link count 1.
        call insert(body_t, head_t)
    end do
```

Example 8.2

Let us consider the database table 7.1. The frequent items are 2, 3, 4, 5, 6, 7 and 8. If we sort them in the order of their frequency, then they appear in the order 5, 3, 4, 7, 2, 6, and 8. If transactions are written in terms of only frequent items, then the transactions are

5, 6, 8
4, 2, 8
5, 4, 7
3
5, 7, 6
3, 4, 2
7, 2, 6
5
8
5, 3, 4, 7
5, 3, 4, 7
5, 6, 8
3, 4, 7, 2, 6
5, 3, 4, 7
3, 2.

The scan of the first transaction leads to the construction of the first branch of the tree (Figure 8.2). The items are ordered in the same way as the transaction appears in the database. The items are ordered according to the decreasing order of frequency of the frequent items. The complete table is given in Figure

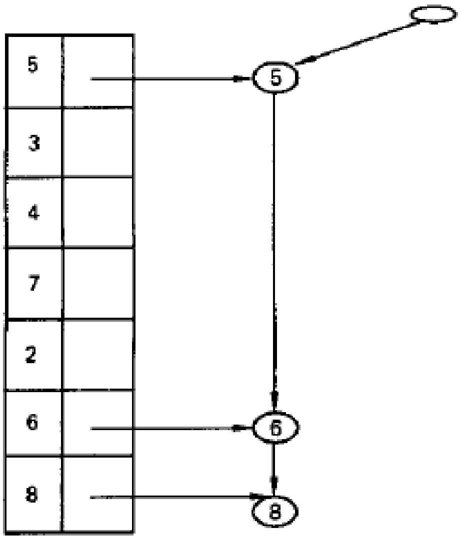


Figure 8.2 Illustration of insert (*t*, root) operation

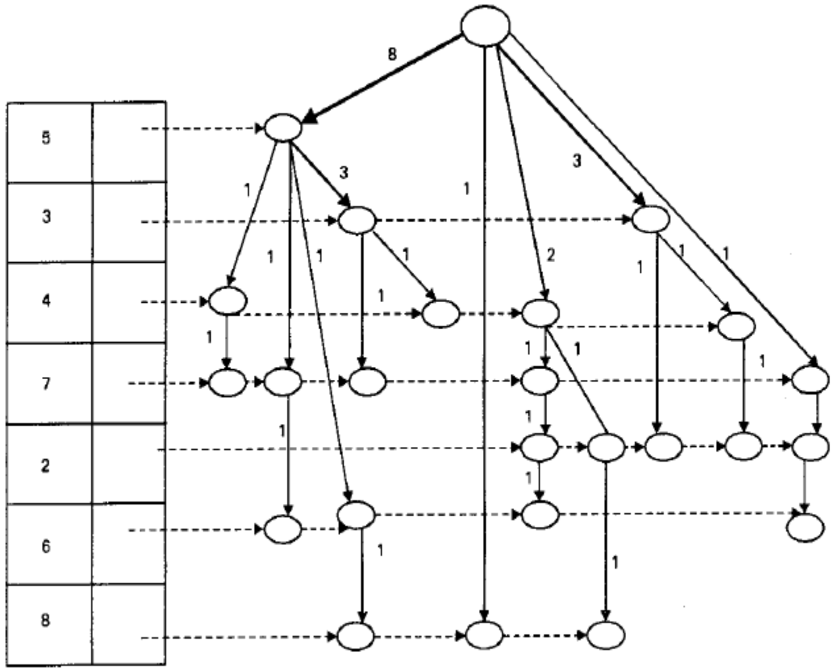


Figure 8.3 Complete FP-tree of the above example. Labels on edges represent frequency.

The algorithm starts from the leaf node in a bottom-up approach.

Thus, after processing item 6, it processes item 7.

Let us consider the frequent item {6}. There are four paths to 6

from the root; these are {5, 7, 6}, {5, 6}, {4, 7, 2, 6} and {7, 2, 6}. All these paths have labels of 1. A label of a path is the smallest of the link counts of its links. Thus, each of these combinations appears just once. The paths {5, 7, 6}, {5, 6}, {4, 7, 2, 6} and {7, 2, 6} from the root to the nodes with label 6 are called the prefix subpaths of 6.

The prefix subpath of a node *a* are the path from the root to the nodes labeled *a* and the count of links along a path are adjusted by adjusting the frequency count of every link in such a path, so that they are the same as the count of the link incident on *a* along the path. This is called a transformed prefix path.

The transformed prefix paths of a node *a* form a truncated database of patterns which co-occur with *a*. This is called a conditional pattern base.

Once the conditional pattern base is derived from the FP-tree, one can compute all the frequent patterns associated with it in the conditional pattern base. By creating a small conditional FP-tree for *a*, the process is recursively carried out starting from the leaf nodes.

The conditional pattern base for {6} is the following:

For the prefix subpath {5, 7, 6}, we get {5, 6} →1, {7, 6}→1, {5, 7, 6}→1.

For the prefix subpath {5, 6}, we get {5, 6}→1.

For the path{4, 7, 2, 6}, we have {4, 6}, {7, 6}, {2, 6}, {4, 7, 6}, {4, 2, 6}, {7, 2, 6}, {4, 7, 2, 6} and for the path {7, 2, 6}, {7, 6}, {2, 6}, {7, 2, 6} all with label 1.

In Figure 8.4 the conditional pattern base of 7 is illustrated. Since the processing is bottom-up, the combination {6, 7} is already considered when the item 6 was considered.

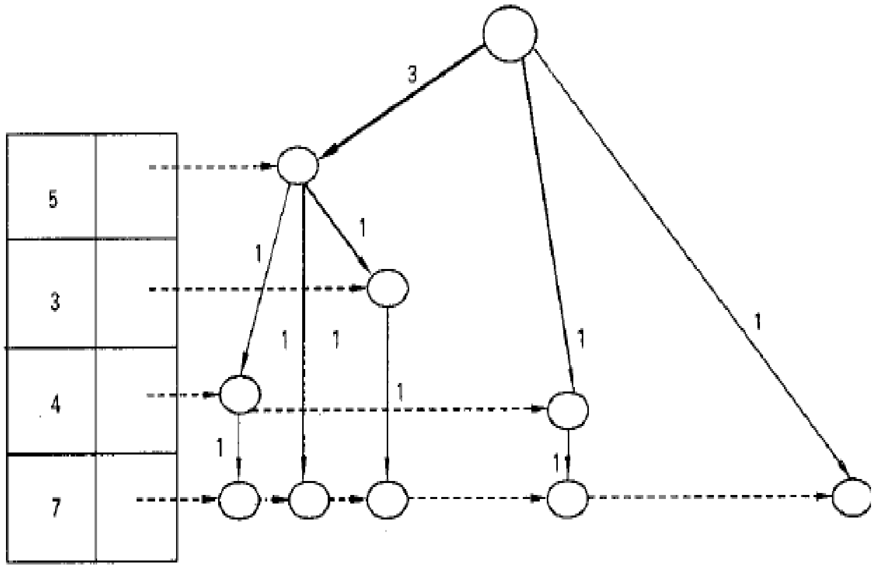


Figure 8.4 Conditional pattern base for item 7

8.5 Check your progress questions

1. How DIC is compared to the working of train?
2. What is FP-Tree?
3. What are the four different structures used in DIC?

8.6 Answer to check your progress questions

1. Dynamic itemset Counting works like a train running over the data, with stops at intervals M between transactions. When the train reaches the end of the transaction file, it has made one pass over the data, and it starts all over again from the beginning for the next pass. The passengers on the train are itemsets. DIC is flexible in allowing itemsets to get on at any stop as long as they get off at the same stop the next time the train goes around. Therefore, the itemset has seen all the transactions in the file.
2. A frequent pattern tree (FP-tree) is a tree structure consisting of an item-prefix-tree and a frequent-item-header table.

   Item-prefix-tree:

   - It consists of a root node labeled null
   - Each non-root node consists of three fields: Item name,
   - support count, and node link.

Frequent-item-header-table:

- Item name;
- Head of node link which points to the first node in the FP-Tree carrying the item name.

3. Four different structures in DIC are:
   - Dashed Box
   - Dashed Circle
   - Solid Box
   - Solid Circle

8.7 Summary

Dynamic Itemset Counting (DIC) algorithm was proposed by Bin *et al*. in 1997. FP-Tree Growth algorithm was proposed by Han *et al*.In A priori algorithm, all itemsets get on the start of a pass and get off at the end. The 1-itemsets take the first pass, the 2-itemsets take the second pass, and so on. In DIC, there is a flexibility of allowing itemsets to get on at any stop as long as they get off at the same stop the next time the train goes around. The FP-Tree Growth algorithm involves two phases. In Phase I, it constructs the FP-tree with respect to a givenσ. The construction of this tree requires two passes over the whole database. In Phase II, the algorithm does not use the transaction database anymore, but it uses the FP-tree.

8.8 Keywords

Frequent Pattern Tree – It constructs the FP-tree with respect to a given σ. The construction of this tree requires two passes over the whole database. In Phase II, the algorithm does not use the transaction database anymore, but it uses the FP-tree.

Transformed prefix path - The prefix subpath of a node *a* are the path from the root to the nodes labeled *a* and the count of links along a path are adjusted by adjusting the frequency count of every link in such a path, so that they are the same as the count of the link incident on *a* along the path. This is called a transformed prefix path.

Conditional pattern base - The transformed prefix paths of a node *a* form a truncated database of patterns which co-occur with *a*. This is called a conditional pattern base.

8.9 Self Assessment Questions and Exercises

1. Explain in detail about DIC algorithm.
2. Explain in detail about FP-Tree algorithm.

8.10 Further Reading

1. Arun K Pujari, Data Mining Techniques, Universities Press
2. Poonkuzhali. S, Saravanakumar. C, Data Warehousing & Data Mining, Charulatha Publications.
3. Jiawei Han, Micheline Kambar, Jian Pei, Data mining concepts and techniques, Morgan Kaufmann is an imprint of Elsevier.
4. Bharat Bhushan Agarwal, Sumit Prakash Tayal, DataMining and Data Warehousing, University Science Press.
5. https://www.techopedia.com/definition/30306/association-rule-mining
6. http://www.ijcsonline.com/IJCS/IJCS_2018_0303010.pdf

UNIT – 9
CLASSIFICATION

Structure

9.1 Introduction

A Classification is a form of data analysis that extracts models describing important data classes. Such models, called classifiers, predict categorical (discrete, unordered) class labels. The classification has numerous applications, including fraud detection, target marketing, performance prediction, manufacturing, and medical diagnosis. Classification predicts categorical class (discrete, unordered) labels. It classifies the data (constructs a model) based on the training set and the values (class labels) in classifying the attribute and uses it in classifying the new data.

9.2 Objective

After going through the unit you will be able to:

- Understand a technique in data mining i.e., classification
- Gain knowledge about the classification algorithm in detail.
- Discuss the decision tree classification.

- Know about Bayesian classification and, Finally about classification by back propagation.

9.3 Decision Tree Classification

Decision tree induction is the learning of decision trees from class-labeled training tuples. A decision tree is a flowchart-like tree structure, where each internal node (non-leaf node) denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (or terminal node) holds a class label. The topmost node in a tree is the root node. A typical decision tree is shown in Figure 9.1.

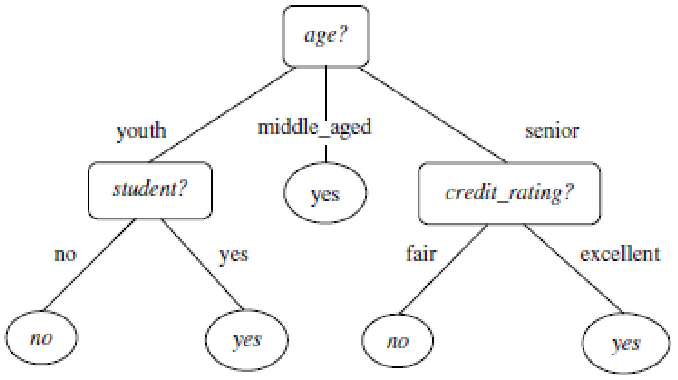


Figure 9.1 A decision tree for the concept whether an All Electronics customer is likely to purchase a computer. Each internal node represents a test on an attribute. Each node represents a class(whether the customer buy a computer or not).

Reason for using decision trees in classification:

- Given a tuple, X, for which the associated class label is unknown, the attribute values of the tuple are tested against the decision tree.
- A path is traced from root to leaf node, which holds the class prediction for that tuple.
- Decision trees can be easily converted to classification rules.

Features of decision tree

The construction of decision tree classifiers does not require any domain knowledge or parameter setting, and therefore is appropriate for exploratory knowledge discovery.

- Decision trees can handle high dimensional data.
- Their representation of acquired knowledge in tree form is intuitive.

- Easy to assimilate by humans.
- The learning and classification steps of decision tree induction are simple and fast.
- Decision tree classifiers have good accuracy.

Decision tree Algorithm

If the tuples in D are all of the same class, then node N becomes a leaf and is labeled with that class (steps 2 and 3). Note that steps 4 and 5 are terminating conditions. All terminating conditions are explained at the end of the algorithm.

Otherwise, the algorithm calls Attribute selection method to determine the splitting criterion. The splitting criterion tells us which attribute to test at node N by determining the "best" way to separate or partition the tuples in D into individual classes (step 6). The splitting criterion also tells us which branches to grow from node N with respect to the outcomes of the chosen test. The node N is labeled with the splitting criterion, which serves as a test at the node (step 7).

Algorithm: Generate_decision_tree. Generate a decision tree from the training tuples of data partition, *D*.

Input:

- Data partition, *D*, which is a set of training tuples and their associated class labels;
- *attribute_list*, the set of candidate attributes;
- *Attribute_selection_method*, a procedure to determine the splitting criterion that "best" partitions the data tuples into individual classes. This criterion consists of a *splitting_attribute* and, possibly, either a *split-point* or *splitting subset*.

Output: A decision tree.

Method:

```
(1)  create a node N;
(2)  if tuples in D are all of the same class, C, then
(3)      return N as a leaf node labeled with the class C;
(4)  if attribute_list is empty then
(5)      return N as a leaf node labeled with the majority class in D; // majority voting
(6)  apply Attribute_selection_method(D, attribute_list) to find the "best" splitting_criterion;
(7)  label node N with splitting_criterion;
(8)  if splitting_attribute is discrete-valued and
         multiway splits allowed then // not restricted to binary trees
(9)      attribute_list ← attribute_list − splitting_attribute; // remove splitting_attribute
(10) for each outcome j of splitting_criterion
     // partition the tuples and grow subtrees for each partition
(11)     let Dj be the set of data tuples in D satisfying outcome j; // a partition
(12)     if Dj is empty then
(13)         attach a leaf labeled with the majority class in D to node N;
(14)     else attach the node returned by Generate_decision_tree(Dj, attribute_list) to node N;
     endfor
(15) return N;
```

A branch is grown from node N for each of the outcomes of the splitting criterion. The tuples in D are partitioned accordingly (steps
9 to 11). Let A be the splitting attribute. A has *v* distinct values,
10 $\{a_1, a_2, \ldots a_v\}$, based on the training data.
The algorithm uses the same process recursively to form a decision tree for the tuples at each resulting partition, D_j , of D (step 14). The recursive partitioning stops only when any one of the following terminating conditions is true:

1. All the tuples in partition D (represented at node N) belong to the same class (Steps 2 and 3).
2. There are no remaining attributes on which the tuples may be further partitioned (step 4). In this case, majority voting is employed (step 5). This involves converting node N into a leaf and labeling it with the most common class in D. Alternatively, the class distribution of the node tuples may be stored.
3. There are no tuples for a given branch, that is, a partition Dj is empty (step 12). In this case, a leaf is created with the majority class in D (step 13). The resulting decision tree is returned (step 15).

Example:

RID	*age*	*income*	*student*	*credit_rating*	*Class: buys_computer*
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Figure 9.2 Class-labeled Training Tuples from the AllElectronics Customer database

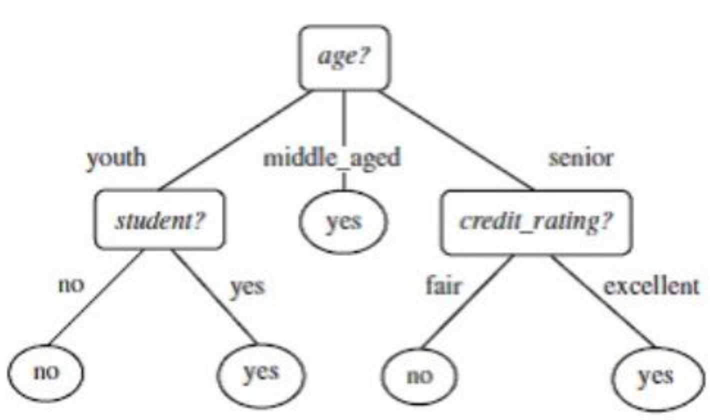


Figure 9.3 Final Decision tree diagram for Figure 9.2

9.4 Bayesian Classification

Bayesian classification is based on Bayes' Theorem. Bayesian classifiers are the statistical classifiers. Bayesian classifiers can predict class membership probabilities such as the probability that a given tuple belongs to a particular class. Bayesian classifiers have also exhibited high accuracy and speed when applied to large databases.

- **Naïve Bayesian classifiers** assume that the effect of an attribute value on a given class is independent of the values of the other attributes.
- **Bayesian belief network**s are graphical models, which unlike naïve Bayesian classifiers allow the representation of dependencies among subsets of attributes.

Baye's Theorem

Bayes' Theorem is named after Thomas Bayes. It plays a critical role in a probabilistic learning and classification. Let *X* be a data tuple whose class label is unknown. Let *H* be some hypothesis, such as that the data tuple *X* belongs to specified class C. Then the classification problem is determined by *P (H/X)*, the probability that the hypothesis *H* holds given the "evidence" or observed data tuple *X*.

There are two types of probabilities:

- Posterior Probability [P(H/X)]
- Prior Probability [P(H)]
-

According to Bayes' Theorem,

$$P(H/X)= P(X/H)P(H) / P(X) \quad (9.1)$$

Bayesian Belief Network

Bayesian Belief Networks specify joint conditional probability distributions. They are also known as Belief Networks, Bayesian Networks, or Probabilistic Networks.

- A Belief Network allows class conditional independencies to be defined between subsets of variables.
- It provides a graphical model of the causal relationship in which learning can be performed.
- We can use a trained Bayesian Network for classification.

Two components define a Bayesian Belief Network

- Directed acyclic graph
- A set of conditional probability tables

Directed Acyclic Graph

- Each node in a directed acyclic graph represents a random variable.
- These variables may be discrete or continuous valued.
- These variables may correspond to the actual attribute given in the data.

Directed Acyclic Graph Representation

The following diagram shows a directed acyclic graph for six Boolean variables.

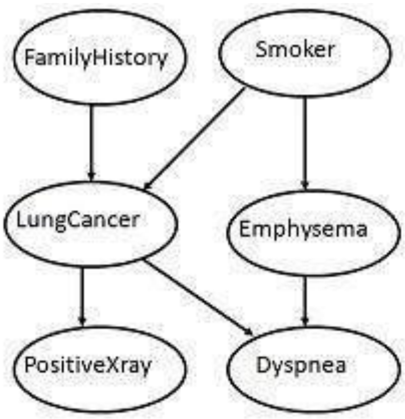


Figure 9.4 A simple Bayesian belief network with directed acyclic graph.

The arc in the diagram allows the representation of causal knowledge. For example, lung cancer is influenced by a person's family history of lung cancer, as well as whether or not the person is a smoker. It is worth noting that the variable Positive X-ray is independent of whether the patient has a family history of lung

cancer or that the patient is a smoker, given that we know the patient has lung cancer.

Conditional Probability Table

The conditional probability table for the values of the variable Lung Cancer (LC) showing each possible combination of the values of its parent nodes, Family History (FH), and Smoker (S) is as follows:

	FH,S	FH,-S	-FH,S	-FH,S
LC	0.8	0.5	0.7	0.1
-LC	0.2	0.5	0.3	0.9

Figure 9.4 A simple Bayesian belief network with the conditional probability table.

Example for Predicting a class label using naive Bayesian classification

Let us predict the class label of a tuple using naıve Bayesian classification, given the same training data as in Figure 9.2 for decision tree induction. The training data were shown earlier in Figure 9.2.

The data tuples are described by the attributes *age*, i*ncome*, *student*, and *credit rating*. The class label attribute, *buys_computer*, has two distinct values (namely, {*yes*, *no*}). Let *C*1 correspond to the class *buys_computer=yes* and *C*2 correspond to *buys_computer= no.*

The tuple we wish to classify is
X = (*age* = *youth*, *income* = *medium*, *student* = *yes*, *credit_rating* = *fair*)

We need to maximize $P(\boldsymbol{X} | C_i) P(C_i)$, for $i = 1, 2$. $P(C_i)$, the prior probability of each class, can be computed based on the training tuples:

P(*buys_computer* D *yes*) = 9/14 = 0.643
P(*buys_computer* D *no*) = 5/14 = 0.357

To compute $P(\boldsymbol{X} | C_i)$, for $i = 1, 2$, we compute the following conditional probabilities:
P(*age* = *youth* | *buys_computer* = *yes*) = 2/9 = 0.222
P(*age* = *youth* | *buys_computer* = *no*) = 3/5 = 0.600
P(*income* =*medium* | *buys_computer* = *yes*) = 4/9 = 0.444
P(*income* = *medium* | *buys_computer* = *no*) = 2/5 = 0.400
P(*student* = *yes* |*buys_computer* = *yes*) = 6/9 = 0.667
P(*student* = *yes* | *buys_computer* = *no*) = 1/5 = 0.200

$P(credit\ rating = fair \mid buy_\ computer = yes) = 6/9 = 0.667$
$P(credit\ rating = fair \mid buys\ computer = no) = 2/5 = 0.400$

Using these probabilities, we obtain

$$\begin{aligned} P(X \mid buys_computer = yes) &= P(age = youth \mid buys_computer = yes) \\ &\times P(income = medium \mid buys_computer = y \\ &\times P(student = yes \mid buys_computer = yes) \\ &\times P(credit_rating = fair \mid buys_computer = \\ &= 0.222 \times 0.444 \times 0.667 \times 0.667 = 0.044. \end{aligned}$$

Similarly,

$$P(X \mid buys_computer = no) = 0.600 \times 0.400 \times 0.200 \times 0.400 = 0.01$$

To find the class, C_i , that maximizes $P(\boldsymbol{X} \mid C_i)\ P(C_i)$, we compute

$$P(X \mid buys_computer = yes) P(buys_computer = yes) = 0.044 \times 0.643 = 0.0$$
$$P(X \mid buys_computer = no) P(buys_computer = no) = 0.019 \times 0.357 = 0.00$$

Therefore, the naive Bayesian classifier predicts *buys_computer* = *yes* for tuple ***X***.

9.5 Classification by Back Propagation

A neural network is a set of connected input/output units in which each connection has a weight associated with it. During the learning phase, the network learns by adjusting the weights so as to be able to predict the correct class label of the input tuples. Neural network learning is also referred to as connectionist learning due to the connections between units.

Each neuron receives a signal from the neurons in the previous layer, and each of those signals is multiplied by a separate weight value-added. The weighted inputs are summed and passed through a limiting function which scales the output to a fixed range of values. The output of the limiter is then, broadcast to all of the neurons in the next layer. So, to solve the problem using this network, we apply the input values to the inputs of the first layer, allow the signals to propagate via the network, and read the output values.

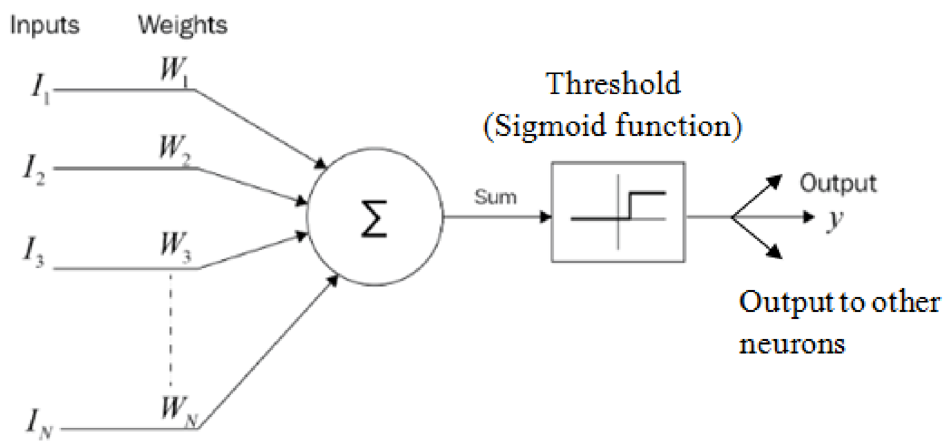


Figure 9.5 The structure of a Neuron

A Disadvantage of Neural networks:

- Involves long training times.
- Poor interpretability
- Verification difficult

Advantages of neural networks:

- High Tolerance for noisy data
- Continue learning
- Ability to classify patterns on which they have not been trained
- Easy parallelization

A Multilayer Feed-Forward Neural Network

The back propagation algorithm performs learning on a multilayer feed-forward neural network. It iteratively learns a set of weights for prediction of the class label of tuples. A multilayer feed-forward neural network consists of an input layer, one or more hidden layers, and an output layer.

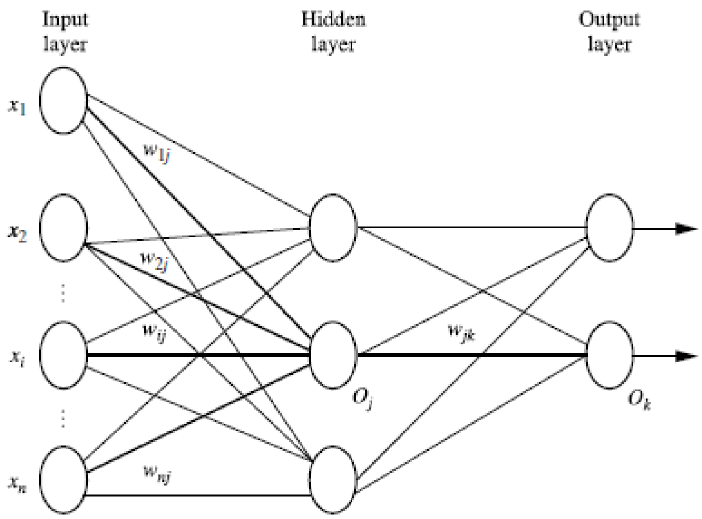


Figure 9.6 A multilayer feed-forward neural network.

Each layer is made up of units. The inputs to the network correspond to the attributes measured for each training tuple. The inputs are fed simultaneously into the units making up the input layer. These inputs pass through the input layer and are then weighted and fed simultaneously to a second layer of "neuron like" units, known as a hidden layer. The outputs of the hidden layer units can be input to another hidden layer, and so on. The number of hidden layers is arbitrary, although in practice, usually only one is used. The weighted outputs of the last hidden layer are input to units making up the output layer, which emits the network's prediction for given tuples.

The units in the input layer are called input units. The units in the hidden layers and output layer are sometimes referred to as neurodes, due to their symbolic biological basis, or as output units. The multilayer neural network shown has two layers of output units. Therefore, we say that it is a two-layer neural network. (The input layer is not counted because it serves only to pass the input values to the next layer.) Similarly, a network containing two hidden layers is called a three-layer neural network, and so on. It is a feed-forward network since none of the weights cycles back to an input unit or to a previous layer's output unit. It is fully connected in that each unit provides input to each unit in the next forward layer.

Multilayer feed-forward neural networks are able to model the class prediction as a nonlinear combination of the inputs. From a statistical point of view, they perform nonlinear regression. Multilayer feed-forward networks, given enough hidden units and enough training samples, can closely approximate any function.

Defining a Network Topology

The network topology is designed by specifying the number of units in the input layer, the number of hidden layers (if more than one), the number of units in each hidden layer, and the number of units in the output layer. Normalizing the input values for each attribute measured in the training tuples will help speed up the learning phase. Typically, input values are normalized so as to fall between 0.0 and 1.0. Discrete-valued attributes may be encoded such that there is one input unit per domain value.

Network design is a trial-and-error process and may affect the accuracy of the resulting trained network. The initial values of the weights may also affect the resulting accuracy. Once a network has been trained and its accuracy is not considered acceptable, it is common to repeat the training process with a different network topology or a different set of initial weights. Cross-validation techniques for accuracy estimation can be used to help decide when an acceptable network has been found.

Backpropagation

Backpropagation learns by iteratively processing a data set of training tuples, comparing the network's prediction for each tuple with the actual known target value or class label. For each training tuple, the weights are modified so as to minimize the mean-squared error between the network's prediction and the actual target value or actual class.

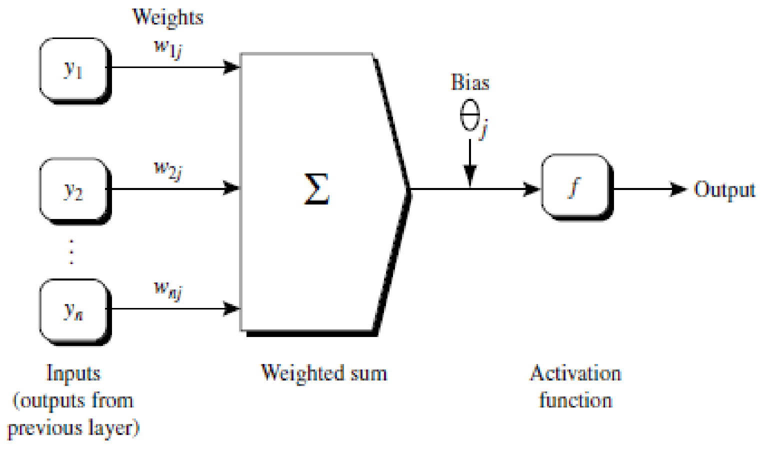


Figure 9.7 Backpropagation network

A major disadvantage of neural networks lies in their knowledge representation. Backpropagation extracts the knowledge embedded in trained neural networks and represents knowledge symbolically for humans to interpret easily. This neural network includes methods for extracting rules from networks and sensitivity analysis.

Algorithm: Backpropagation. Neural network learning for classification or numeric prediction, using the backpropagation algorithm.

Input:

- *D*, a data set consisting of the training tuples and their associated target values;
- *l*, the learning rate;
- *network*, a multilayer feed-forward network.

Output: A trained neural network.

Method:

```
Initialize all weights and biases in network;
while terminating condition is not satisfied {
    for each training tuple X in D {
        // Propagate the inputs forward:
        for each input layer unit j {
            O_j = I_j; // output of an input unit is its actual input value
        for each hidden or output layer unit j {
            I_j = Σ_i w_ij O_i + θ_j; //compute the net input of unit j with respe
                the previous layer, i
            O_j = 1/(1+e^(-I_j)); } // compute the output of each unit j
        // Backpropagate the errors:
        for each unit j in the output layer
            Err_j = O_j(1 − O_j)(T_j − O_j); // compute the error
        for each unit j in the hidden layers, from the last to the first hidden la
            Err_j = O_j(1 − O_j) Σ_k Err_k w_jk; // compute the error with respe
                the next higher layer, k
        for each weight w_ij in network {
            Δw_ij = (l)Err_j O_i; // weight increment
            w_ij = w_ij + Δw_ij; } // weight update
        for each bias θ_j in network {
            Δθ_j = (l)Err_j; // bias increment
            θ_j = θ_j + Δθ_j; } // bias update
        } }
```

9.6. Check your progress questions

1. Define Classification.
2. Define Decision Tree.
3. Define Naïve Bayesian classification.
4. Define Backpropagation.

9.7 Answer to check your progress questions

1. Classification is a form of data analysis that extracts models describing important data classes. Such models, called classifiers, predict categorical (discrete, unordered) class labels.

2. A decision tree is a flowchart-like tree structure, where each internal node (non-leaf node) denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (or terminal node) holds a class label. The topmost node in a tree is the root node.
3. Naive Bayesian classification is based on Bayes' theorem of posterior probability. It assumes class-conditional independence, that the effect of an attribute value on a given class is independent of the values of the other attributes.
4. Backpropagation is a neural network algorithm for classification that employs a method of gradient descent. It searches for a set of weights that can model the data so as to minimize the mean-squared distance between the network's class prediction and the actual class label of data tuples.

9.8 Summary

Classification is a form of data analysis that extracts models describing data classes. Decision tree induction is a top-down recursive tree induction algorithm, which uses an attribute selection measure to select the attribute tested for each non-leaf node in the tree. Naive Bayesian classification is based on Bayes' theorem of posterior probability. Backpropagation is a neural network algorithm for classification. It searches for a set of weights that can model the data so as to minimize the mean-squared distance between the network's class prediction and the actual class label of data tuples.

9.9 Keywords

- **Bayesian belief network** – They are graphical models, which unlike naïve Bayesian classifiers allow the representation of dependencies among subsets of attributes.
- **Neural Network** - It is a set of connected input/output units in which each connection has a weight associated with it.
- **Decision tree induction** – It is the learning of decision trees from class-labeled training tuples.

9.10 Self Assessment Questions and Exercises

1. Briefly explain about Decision tree induction algorithm.
2. Write in detail about Bayesian classification.
3. Discuss classification by Back propagation.

9.11 Further Reading

1. Arun K Pujari, Data Mining Techniques, Universities Press
2. Poonkuzhali. S, Saravanakumar. C, Data Warehousing & Data Mining, Charulatha Publications.
3. Jiawei Han, Micheline Kambar, Jian Pei, Data mining concepts and techniques, Morgan Kaufmann is an imprint of Elsevier.
4. Bharat Bhushan Agarwal, Sumit Prakash Tayal, Data Mining and Data Warehousing, University Science Press.
5. https://www.techopedia.com/definition/30306/association-rule-mining
6. http://www.ijcsonline.com/IJCS/IJCS_2016_0303010.pdf

BLOCK - 4
CLUSTERING TECHNIQUES

UNIT – 10
INTRODUCTION TO CLUSTERING

Structure

10.1 Introduction

Clustering is a useful technique for the discovery of data distribution and patterns in the underlying data. The goal of clustering is to discover both the dense and the sparse regions in a

data set. Data clustering has been studied in statistics, machine learning, etc. The basic principle of clustering hinges on the

concept of distance metric or similarity metric.

10.2 Objective

After going through the unit you will be able to:

- Understand the clustering algorithm for numerical data
- Understand the clustering algorithm for categorical data
- Principles of partitioning algorithm and three different partitioning algorithms
- Know the concepts of hierarchical clustering algorithms
- Understand the concept of BIRCH

10.3 Clustering Paradigms

There are two main approaches to clustering – hierarchical clustering and partitioning clustering. Clustering algorithms differ

among themselves in their ability to handle different types of attributes, numeric and categorical, and in accuracy of clustering.

10.3.1 Hierarchical Vs Partitioning

The partition clustering techniques partition the database into a predefined number of clusters. The partition clustering algorithms are of two types: k-means algorithms and k-medoid algorithms. The hierarchical clustering techniques do a sequence of partitions, in which each partition is nested into the next partition in the sequence. It creates a hierarchy of clusters from small to big or big to small. The hierarchical techniques are of two types: agglomerative and divisive clustering techniques.Agglomerative clustering techniques start with as many clusters as there are records, with each cluster having only one record. At each stage, the pairs of the clusters that are merged are the ones nearest to each

other. If the merging is continued, it terminates in a hierarchy of clusters which is built with just a single cluster containing all the records, at the top of the hierarchy. Divisive clustering techniques take the opposite approach from agglomerative techniques. This starts with all the records in one cluster, and then tries to split that cluster into small pieces.

10.3.2 Numeric Vs Categorical

The clustering of numerical data, geometric properties can be used to define the distances between the points. In, the clustering of categorical data such a criterion does not exist and many data sets also consist of categorical attributes, on which distance functions are not naturally defined. It is hard to determine an ordering or to quantify the dissimilarity among categorical attributes.

10.4 Partitioning Algorithm

Partitioning algorithms construct partitions of a database of *N* objects into a set of *k* clusters. The partitioning clustering algorithm adopts the iterative optimization paradigm. It starts with an initial partition and uses an iterative control strategy.

There are the two main categories of partitioning algorithms.

i. *k*-means algorithms, where each cluster is represented by the center of gravity of the cluster.
ii. *k*-medoid algorithms, where each cluster is represented by one of the objects of the cluster located near the center.

10.5 *k*-Means Algorithm

The *k*-means algorithm is an evolutionary algorithm that gains its name from its method of operation. The algorithm clusters observations into *k* groups, where *k* is provided as an input parameter. It then assigns each observation to clusters based upon the observation's proximity to the mean of the cluster. The cluster's mean is then recomputed and the process begins again. Here's how the algorithm works:

1. The algorithm arbitrarily selects *k* points as the initial cluster centers (the means).
2. Each point in the dataset is assigned to the closed cluster, based upon the Euclidean distance between each point and each cluster center.
3. Each cluster center is recomputed as the average of the points in that cluster.
4. Steps 2 and 3 repeats until the clusters converge. Convergence may be

defined differently depending upon the implementation, but

it normally means that either no observations change

clusters when steps 2 and 3 are repeated, or that the changes

do not make a material difference in the definition of the clusters.

Disadvantages of k-means clustering algorithms

One of the main disadvantages of the *k*-means clustering is the fact that you must specify the number of clusters as an input to the algorithm. As designed, the algorithm is not capable of determining the appropriate number of clusters and depends upon the user to identify this in advance.For example, if you had a group of people that are to be clustered based upon binary gender identity as male or female, calling the *k*-means algorithm using the input *k=3* would force the people into three clusters when only two, or input of *k=2,* would provide a more natural fit. Similarly, if a group of individuals was easily clustered based upon home state and you called the *k*-means algorithm with the input *k=20,* the results might be too generalized to be effective.

10.6 *k*-Medoid Algorithm

10.6.1 PAM

PAM (Partition Around Medoids) uses a *k*-medoid method to identify the clusters. PAM selects *k* objects from the data as medoids. Each of these *k* objects is representative of *k* classes.
Other objects in the database are classified based on their distances to these *k*-medoids.
The algorithm starts with arbitrarily selected *k*-medoids and iteratively improves upon this selection. In each step, a swap between a selected object O_i and a non-selected object O_h is made, as long as such a swap results in an improvement in the quality of clustering.
To calculate the effect of such a swap between O_i and O_h a cost C_{ih} is computed, which is related to the quality of partitioning the non-selected objects to *k* clusters represented by the medoids.
The algorithm has two important modules: the partitioning of the database for a given set of medoids and the iterative selection of medoids.

10.6.2 Partitioning

If O_j is a non-selected object and O_i is a medoid, we then say that O_j belongs to the cluster represented by O_i, if $d(O_i, O_j) = Min_{O_e} d(O_j, O_e)$, where the minimum is taken over all medoids O_e

and $d(O_a, O_b)$ determines the distance, or dissimilarity, between

objects O_a and O_b. The quality of clustering is measured by the average dissimilarity between an object and the medoid of the cluster to which the object belongs.

10.6.3 Iterative Selection of Medoids

The following figures (Figures 10.1 and 10.2) illustrate the working of the swapping in PAM. There are 12 objects which are required to be classified into 4 clusters.

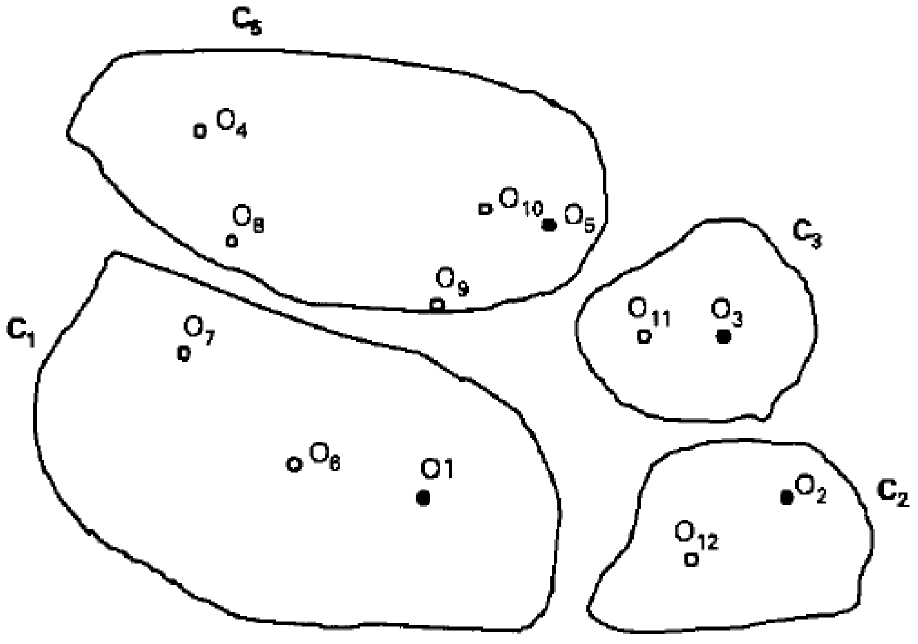


Figure 10.1 Partitioning Before Swapping

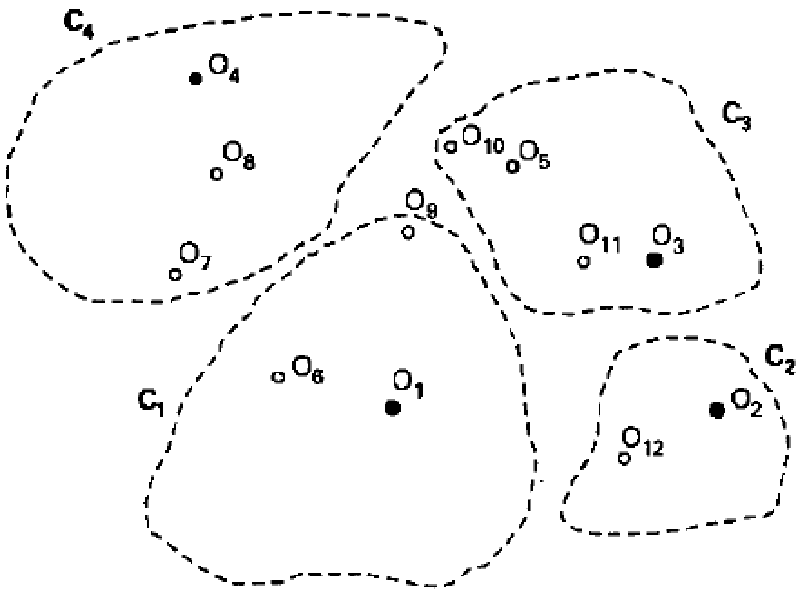


Figure 10.2 Clustering after swapping O_5 by O_4

Initially, the selected medoids are O_1, O_2, O_3 and O_5. Based on closest distance, the remaining objects are classified as shown in the figures. Let us assume that O_4 is introduced as a medoid in place of O_5. For convenience, we denote d_{ij} for $d(O_i, O_j)$ and the distance metric is symmetric. The three cases are illustrated below.

Case 1 The object O_8 which was in the cluster C_5 before swapping is no\ cluster C_4 after swapping.

Hence, the cost $C_{854} = d_{58} - d_{48}$.

Case 2 The object O_9 was in C_5 before swapping and is now in C_1 after swap|

The cost for O_9 is $C_{954} = d_{19} - d_{59}$.

Case 3 The object O_7 was in the cluster C_1 before swapping and is in ‹ swapping.

Thus, the cost is $C_{754} = d_{47} - d_{17}$.

Define the total cost of swapping O_i and O_h as

$$C_{ih} = \sum_j C_{jih},$$

where the sum is taken for all objects j.

The algorithm can be formally stated as follows:

PAM Algorithm

Input: Database of objects D.
select arbitrarily k representative objects, *Kmed*.
mark these objects as "selected" and mark the remaining as "non-selected".
    do for all selected object O_i
        do for all non-selected objects O_h
            compute C_{ih}
        end do.
    end do.

select i_{min}, h_{min} such that $C_{i_{min}, h_{min}} = \text{Min}_{i,h}\, C_{ih}$
    if $C_{i_{min}, h_{min}} < 0$
        then swap: mark O_i as non-selected and O_h as selected.
    repeat
find clusters $C_1, C_2, C_3, \ldots, C_k$.

PAM is very robust to the existence of outliers. The clusters found by this method do not depend on the order in which the objects are examined. However, it cannot handle very large volumes of data.

10.7 CLARA

It can be observed that the major computational efforts for PAM are to determine *k*-medoids through an iterative optimization. Though CLARA follows the same principle, it attempts to reduce the computational effort. Instead of finding representative objects for the entire data set, CLARA draws a sample of the data set and applies PAM on this sample to determine the optimal set of medoids from the sample.

It then classifies the remaining objects using the partitioning principle. If the sample were drawn in a sufficiently random way, the medoids of the sample would approximate the medoids of the entire data set. The steps of CLARA are summarized.

CLARA Algorithm

Input: Database of D objects.
repeat for m times
    draw a sample $S \subseteq D$ randomly from D.
    call PAM (S, k) to get k medoids.
    classify the entire data set D to $C_1, C_2 \ldots C_k$.
    calculate the quality of clustering as the average dissimilarity.
end.

10.8 CLARANS

CLARANS (Clustering Large Applications based on Randomized Search) is similar to PAM and CLARA, but it applies a randomized Iterative-Optimization for the determination of medoids. Some of the disadvantages of PAM and CLARA are rectified by CLARAN such as, CLARANS which does not restrict the search to any particular subset of objects. Neither does it search the entire data set. It begins with PAM and randomly selects a few pairs (*i, h*), instead of examining all pairs, for swapping at the current state. CLARANS, like PAM, starts with a randomly selected set of *k*-medoids. It checks at most the "maxneighbor" numbers of pairs for swapping and, if a pair with the negative cost is found, it updates the medoid set and continues. Otherwise, it records the current selection of medoids as a local optimum and restarts with a new randomly selected medoid, set to search for another local optimum. CLARANS stops after the "numlocal" number of local optimal medoid sets is determined, and returns the best among these.

CLARANS Algorithm

Input (D, k, maxneighbor and numlocal)
select arbitrarily k representative objects.
mark these objects as "selected" and all other objects as non-selected. Call it current.
set $e = 1$.
do while ($e \leq$ numlocal)
    set $j = 1$
    do while ($m \leq$ maxneighbor)
        select randomly a pair (j, h) such that O_j is a selected object and O_h is a non-select
        compute the cost C_{jh}.
        if C_{jh} is negative
            "update current"
            mark O_j non-selected, O_h selected and $m = 1$
        else
            increment $m \leftarrow m + 1$
    end do
    compare the cost of clustering with "mincost"
    if current_cost < mincost
        mincost ← current_cost
        best_node ← current
    increment $e \leftarrow e + 1$
end do
return "best node"

CLARANS is a medoid-based method, which is more efficient than the earlier medoid-based methods, but suffers from two major drawbacks: it assumes that all objects fit into the main memory, and the result is very sensitive to input order. It may not find a real local minimum due to the trimming of its searching, controlled by "maxneighbor".

10.9 Hierarchical Clustering

Hierarchical algorithms create a hierarchical decomposition of the database. The algorithms iteratively split the database into smaller subsets, until some termination condition is satisfied. The hierarchical algorithms do not need *k* as an input parameter, which is an obvious advantage over partitioning algorithms.

Hierarchical methods group data into a tree of clusters. There are two basic varieties of hierarchical algorithms; agglomerative and divisive. A tree structure called a dendrogram is commonly used to represent the process of hierarchical clustering.

Agglomerative hierarchical methods: Bottom-up strategy: Begin with as many clusters as objects. Clusters are successively merged until only one cluster remains.

Divisive hierarchical methods: Top-down strategy: Begin with all objects in one cluster. Groups are continually divided until there are as many clusters as objects.

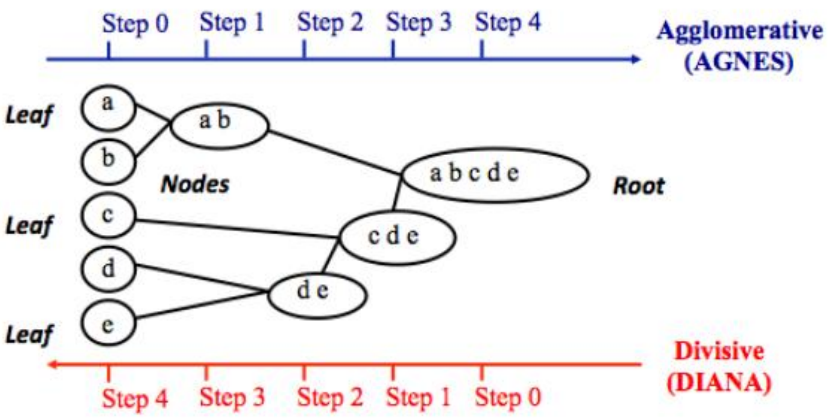


Figure 10.3 Agglomerative and divisive hierarchical clustering on data objects *{a, b, c, d, e}*.

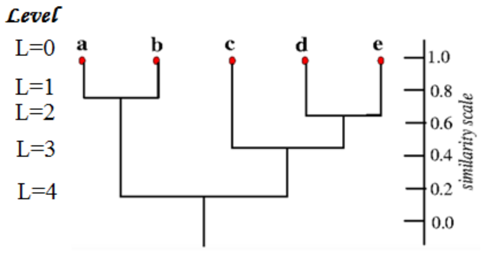


Figure 10.4 Dendrogram representation for hierarchical clustering of data objects ***{a, b, c, d, e}*.**

10.10 DBSCAN

DBSCAN stands for Density-Based Spatial Clustering of Applications with Noise. It consists of two parameters:

- Eps: Maximum radius of the neighborhood
- MinPts: Minimum number of points in the Eps-neighborhood of a point.

The density of a point is defined as the number of points within a specified radius (Eps). A point is a core point if it has more than a specified number of points (MinPts) with Eps. A border point has fewer than MinPts within Eps, but is in the neighborhood of a core point. A noise point is any point that is not a core point or a border point. Any two core points are close enough, within a distance Eps of one another, are put in the same cluster. Any border point that is close enough to a core point is put in the same cluster as the core point. Noise points (outlier) are discarded.

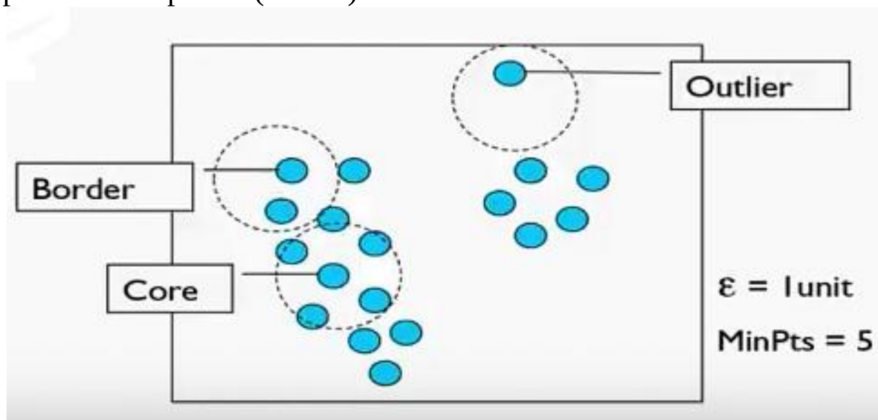


Figure 10.5 Algorithm divides the points into three groups based on density

Density-Reachability is a point *q* is directly density-reachable from a point *p*, if *p* is a core point and *q* is in *p*'s neighborhood.

Density-Connectivity is a pair of points *p* and *q* are density connected, if they are commonly density-reachable from a point *o*.

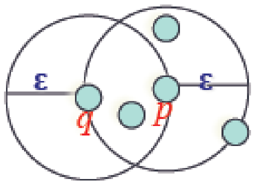
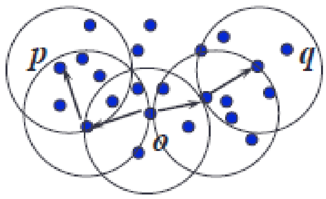


Figure 10.6 Density-Reachability with Minpts=4.



DBSCAN Algorithm

Algorithm DBSCAN (*D*, *ε*, MinPts)
Input: Database of objects *D*
do for all $O \in D$
    if *O* is unclassified
        call function **expand_cluster**(*O*, *D*, *ε*, MinPts)
end do.

Function expand_cluster (*O*, *D*, *ε*, MinPts):
get the *ε*-neighbourhood of *O* as $N_\varepsilon(O)$
    if $|N_\varepsilon(O)|$ < MinPts,
        mark *O* as noise
        return
    else
        select a new cluster_id and mark all objects of $N_\varepsilon(O)$ with this cluster-id and put them into candidate-objects.
        do while candidate-objects is not empty
            select an object from candidate-objects as current_object
            delete current-object from candidate-objects
            retrieve N_ε(current-object)
            if $|N_\varepsilon$(current-object)$|$ ≥ MinPts
                select all objects in N_ε(current-object) not yet classified or marked as noise,
                mark all of the objects with cluster_id,
                include the unclassified objects into candidate-objects
        end do
return.

Figure 10.7 Density-Connectivity with Minpts=7.

At every step, the algorithm starts with an unclassified object and a new cluster-id associated with it. D represents Data set, ε-neighbourhood to see if the neighbourhood is adequately dense or not.

If its density does not exceed the threshold MinPts, then it is marked as a noise object. Otherwise, all the objects that are within its ε-neighbourhood are retrieved and put into a list of candidate objects. These objects may be either unclassified or noise. The neighbourhood cannot contain a classified object.

If the object is a noise object, the current cluster-id is assigned to it. If the object is unclassified, then the current cluster-id is assigned to it and it is included in the list of candidate objects for which the ε-neighbourhood are to be obtained.

The algorithm continues till the list of candidate objects is empty. Thus, one cluster with the given cluster-id is determined. The algorithm repeats this process for other unclassified objects and terminates when all the objects are marked as either classified or noise. The algorithm is described below.

10.11BIRCH

BIRCH stands for Balanced Iterative Reducing and Clustering Using Hierarchies. BIRCH partitions object hierarchically using tree structures and then refine the clusters using other clustering methods. It defines a clustering feature and an associated tree structure that summarizes a cluster. The tree (CF tree) is a height-balanced tree that stores cluster information. BIRCH does not produce spherical clusters and may produce an unintended cluster. These structures help the clustering method achieve good speed and scalability in large databases and also make it effective for the incremental and dynamic clustering of incoming objects.BIRCH applies a multiphase clustering technique: a single scan of the data set yields a basic good clustering, and one or more additional scans can (optionally) be used to further improve the quality. The primary phases are:

Phase I : BIRCH scans the database to build an initial in-memory CF tree, which can be viewed as a multilevel compression of the data that tries to preserve the inherent clustering structure of the data.

Phase II: BIRCH applies a (selected) clustering algorithm to cluster the leaf nodes of the CF tree, which removes sparse clusters as outliers and group's dense clusters into larger ones.

Clustering features a summary of the statistics for a given subcluster, the 0^{th}, 1^{st} and 2^{nd} moments of the subcluster from the statistical point of view. It also registers crucial measurements for computing clusters and utilizes storage efficiently.

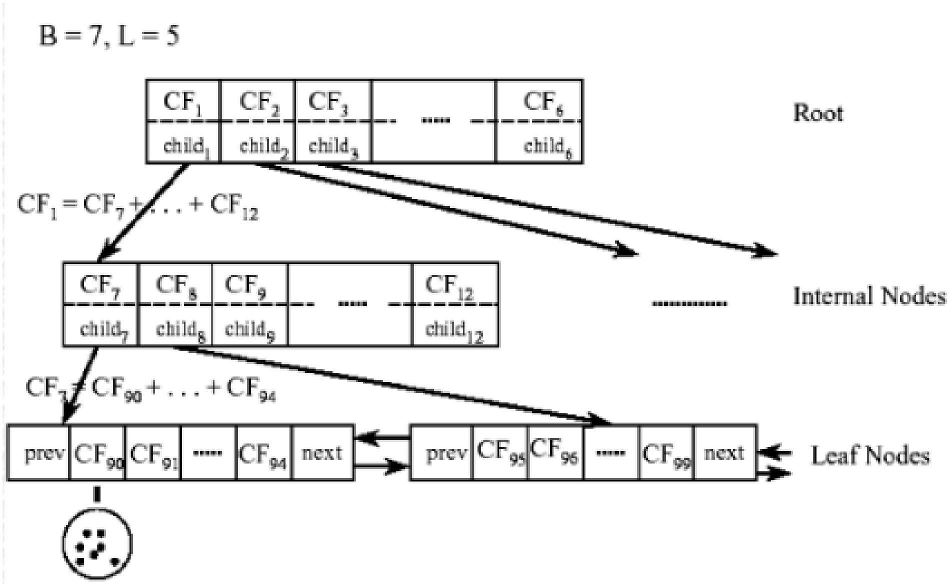


Figure 10.8 CF (Cluster Features) tree

Properties of CF-Tree

- Each non-leaf node has at most B entries.
- Each leaf node has at most L CF entries which each satisfy threshold T.
- Node size is determined by dimensionality of data space and input parameter P (page size).

A CF tree is a height balanced tree that stores the clustering features for a hierarchical clustering. A non leaf node in a tree has descendents or "Children". The non leaf nodes stores sums of the CFs of their children.

A CF tree has two parameters

- Branching factor: specify the maximum number of children
- Threshold: Maximum diameter of sub-clusters stored at the leaf nodes.

Drawback

A CF tree handles only numeric data, and sensitive to the order of the data record.

10.12 CATEGORICAL CLUSTERING ALGORITHMS

Although algorithms like BIRCH, CURE, CLARANS are suitable for large data sets these are designed for numeric data. There have been a few proposals for clustering using a categorical data set. The important algorithms are STIRR, ROCK, and CACTUS. One interesting common feature of these three algorithms is that they attempt to model the similarity of categorical attributes in more or less a similar manner. *A* and *B* are similar if the sets of items with which they co-occur have a large overlap even though these never co-occur together. ROCK tries to introduce a concept called neighbors and links. Two items have link if they have a common neighbor. STIRR defines a weighted node and the weights of co-occurring items will exhibit a similar magnitude of weights. CACTUS also makes use of occurrences as the similarity measure.

10.13 STIRR

STIRR stands for "Sieving Through Iterated Relational Reinforcement." STIRR is an iterative approach for assigning and propagating weights on the categorical values; this allows a similarity measure for categorical data, which is based on the co-occurrence of values in the dataset, STIRR looks for relationships between all attribute values to detect a potential cluster and converges to clusters of highly correlated values between different categorical attribute fields.

Each possible value in a categorical attribute is represented by a node and the data is represented as a set D of objects. Each object $d \in D$ is represented as a set of nodes, consisting of one node from each attribute field. STIRR assigns a weight w_v to each node *v*. The weight configuration is referred to as w. A normalization function $N(w)$ rescales the weights of the nodes associated with an attribute such that their squares add to 1.
A function f is repeatedly applied to a set of nodes (values) until a fixed point u is reached for which $f(u) = u$. The function f maps

the node weights to a new configuration. So, the purpose is to converge to a point that remains the same under repeated applications of *f*. The function f updates a weight w_v for a node *v* by applying an operator ⊕ to all objects that contain value *v* (as well as $m - 1$ other values) and summing the results, as follows:

$for\ each\ object\ o = \{v, u_1 \dots, u_{m-1}\}\ that\ contains\ value\ v$

$$x_r \leftarrow \oplus\ (u_1 \dots, u_{m-1})$$

$$w_v \leftarrow \sum_r x_r$$

The operator ⊕ may be a simple multiplication or addition operator. The function f then normalizes the set of weights using *N*(). After several iterations of yielding a new configuration $f(w)$ the system is expected to converge to a point where $f(w) = w$. Then, the nodes with large positive weights and the nodes with extreme negative weights represent dense regions in the dataset that are separated and have few interconnections, possibly defining clusters.

Disadvantages of STIRR include its sensitivity to the initial object ordering. It also lacks a definite convergence. The notion of weights is nonintuitive and several parameters are user-specified. The final detected clusters are often incomplete.

10.14 ROCK

ROCK stands for Robust Clustering using links). It is a hierarchical clustering algorithm for categorical attributes. The ROCK algorithm is divided into three general parts:

1. Obtaining a random sample of the data.
2. Performing clustering on the data using the link agglomerative approach. A goodness measure is used to determine which pair of points is merged at each step.
3. Using these clusters the remaining data on the disk are assigned to them.

Both local and global heaps are used. The local heap for t_i, $q[t_i]$, contains every cluster that has a nonzero link to $\{t_i\}$. The hierarchical clustering portion of the algorithm starts by placing each point t_i in the sample in a separate cluster. The global heap contains information about each cluster.

All information in the heap is ordered based on the goodness measure:

$$g(K_i, K_J) = \frac{LINK(K_i, K_J)}{(n_i + n_j)^{1+2f(\theta) - n_j^{i+2f(\theta)}}}$$

Here link(K_i, K_J) is the number of links between the two clusters.

Also n_i and n_j are the number of points in each cluster. The function $f(\theta)$ depends on the data, but it is found to satisfy the property that each item in K_i has approximately $n_j^{f(\theta)}$ neighbors in

the cluster.Merges clusters based on their interconnectivity. It is

great for categorical data. ROCK ignores information about the looseness of two clusters while emphasizing interconnectivity.

Steps

Obtain a sample of points from the data set

Compute the link value for each set of points, i.e., transform the

original similarities (computed by Jaccard coefficient) into similarities that reflect the number of shared neighbors between

points.Perform an agglomerative hierarchical clustering on the data

using the "number of shared neighbors" as similarity measure and maximizing "the shard neighbors" objective function.

Assign the remaining points to the clusters that have been found

Link measure in Rock

Links : Number of common neighbors

- C_1 <a, b, c, d, e> : {a, b, c}, {a, b, d}, {a, b, e}, {a, c, d}, {a, c, e}, {a, d, e}, {b, c, d}, { b, c, e}, {c, d, e}
- C_2 <a, b, f, g> : {a, b, f}, {a, b, g}, {a, f, g}, {b, f, g} Let T_1 ={a, b, c}, T_2 ={c, d, e}, T_3 ={a, b, f}
- Link (T_1, T_2) =4, since they have 4 common neighbors. i.e. {a, c, d}, {a, c, e}, {b, c, d}, {b, c, e}
- Link (T_1, T_2) =3, since they have 3 common neighbors. i.e.

{a, b, d}, {a, b, e}, {a, b, g}

10.15 CACTUS

CACTUS stands for Clustering Categorical Data Using Summaries. It was devised by Ganti, Gehrke and Ramakrishnan. The nicety of CACTUS lies in its problem decomposition. Let us assume that the database D is a set of tuples each having k fields or attributes. The clustering techniques are essentially attempting to cluster the tuples by considering the tuple as a primary object.

CACTUS is a projected clustering method, which assumes that a cluster id identified by a unique set of attribute values that seldom occur in other clusters. It searches for the minimal set of relevant attribute sets that are sufficient to define a cluster.

Assume all attributes $A_1, \ldots, A_m$ are independent and all values in an attribute are equally likely. Then, the measure $\sigma(a_i, a_j)$ indicates the co-occurrence (and the similarity) of attribute values a_i and a_j. The values a_i and a_j are strongly connected if their co-occurrence $\sigma(a_i, a_j)$ is higher by a user-specified factor α > 1 than the value expected under the attribute-independence assumption.

A set of attribute values $C = \{a_1, \ldots, a_n\}$ over the attributes $\{A_1, \ldots, A_n\}$ is a cluster if the set C is a set of strongly connected attribute values. In other words, the condition should be satisfied that all pairs of attribute values in C are strongly connected and their co-occurrence $\sigma(a_i, a_j) > \alpha$ for $i \neq j$. Cluster C is also called a subspace cluster. C is a subcluster if there is another value for one of attributes $\{A_1, \ldots, A_n\}$ that is not included in C, but is strongly connected to the other attribute values is $C = \{a_1, \ldots, a_n\}$.

The CACTUS algorithm collects inter-attribute summaries and intra-attribute summaries on categorical attributes. The inter-attribute summaries consist of all strongly connected attribute value pairs where each pair has attribute values from different attributes. The intra-attribute summaries consist of similarities between attribute values of the same attribute. Then, the CACTUS algorithm consists of three phases: summarization, clustering and validation.

The summarization phase computes summary information from the dataset. The clustering phase uses the summary information to discover a set of candidate clusters. The validation phase determines the actual set of clusters from the set of candidate clusters.

A drawback of CACTUS is that the assumption of a cluster being identified by a unique set of attribute values that seldom occur in other clusters may be

unnatural for clustering some real-world datasets. CACTUS may also return too many clusters.

10.16 Check your progress questions

1. Define Clustering.
2. Mention some of the numeric data clustering.
3. Mention some of the categorical data clustering.
4. What are the benefits of CLARA?
5. What are the two important parameters in DBSCAN?

10.17 Answer to check your progress questions

1. Clustering is a useful technique for the discovery of data distribution and patterns in the underlying data. The goal of clustering is to discover both the dense and the sparse
2. regions in a data set.
3. BIRCH, CURE and CLARANS are numeric data clustering.
4. STIRR, ROCK, and CACTUS are categorical data clustering.
5. Though CLARA follows the same principle, it attempts to
6. reduce the computational effort. Instead of finding representative objects for the entire data set, CLARA draws
7. a sample of the data set, and applies PAM on this sample to determine the optimal set of medoids from the sample.
8. DBSCAN stands for Density-Based Spatial Clustering of Applications with Noise. It consists of two parameters:
9. Eps: Maximum radius of the neighborhood MinPts: Minimum number of points in the Eps-neighborhood of a point.

10.18 Summary

The hierarchical clustering techniques do a sequence of partitions, in which each partition is nested into the next partition in the sequence. Partitioning algorithms construct partitions of a database of *N* objects into a set of *k* clusters. The partitioning clustering algorithm adopts the iterative optimization paradigm. It starts with an initial partition and uses an iterative control strategy. Agglomerative clustering techniques start with as many clusters as

there are records, with each cluster having only one record. Divisive clustering techniques take the opposite approach from agglomerative techniques. This starts with all the records in one cluster, and then tries to split that cluster into small pieces.

10.19 Keywords

- **Core point** - A point is a core point if it has more than a specified number of points (MinPts) with Eps.
- **Border point** - A border point has fewer than MinPts within Eps, but is in neighborhood of a core point.
- **Noise point** - A noise point is any point that is not a core point or a border point.
- **Density-Reachability** - It is a point *q* is directly density-reachable from a point *p*, if *p* is a core point and *q* is in *p*'s neighborhood.
- **Density-Connectivity** - is a pair of points *p* and *q* are density connected, if they are commonly density-reachable from a point *o*.
- **BIRCH -** BIRCH stands for Balanced Iterative Reducing and Clustering Using Hierarchies. BIRCH partitions objects hierarchically using tree structures and then refines the clusters using other clustering methods. It defines a clustering feature and an associated tree structure that summarizes a cluster.
- **PAM** – PAM (Partition Around Medoids) uses a *k*-medoid method to identify the clusters. PAM selects *k* objects from the data as medoids. Each of these *k* objects is representatives of *k* classes. Other objects in the database are classified based on their distances to these *k*-medoids.

10.20 Self Assessment Questions and Exercises

1. Explain about partitioning algorithm.
2. Write short notes on hierarchical clustering and categorical clustering algorithm.
3. Explain about BIRCH.

10.21 Further Reading

1. Arun K Pujari, Data Mining Techniques, Universities Press
2. Poonkuzhali. S, Saravanakumar. C, Data Warehousing & Data Mining, Charulatha Publications.
3. Charu C. Aggarwal, Chandan K. Reddy, Data Clustering: Algorithms and Applications, CRC Press
4. Pang-Ning Tan, Vipin Kumar, Michael Steinbach, Introduction to Data Mining, Pearson.
5. Rao. N. Raghavendra, Global Virtual Enterprises in Cloud Computing

UNIT – 11
MACHINE LEARNING

Structure

11.1 Introduction

Machine learning and data mining use the same key algorithms to discover patterns in the data. Unlike data mining, in machine learning, the machine must automatically learn the parameters of models from the data. Machine learning uses self-learning algorithms to improve its performance at a task with experience over time. It can be used to reveal insights and provide feedback in near real-time.

Machine Learning is a concept that allows the machine to learn from examples and experience, and that too without being explicitly programmed. So instead of writing the code, we feed data to the generic algorithm, and the algorithm/machine builds the logic based on the given data.

Machine learning, for example, can be used to continuously monitor the performance of equipment and events and automatically determine what the norm is and when failures are likely to occur. When new datasets are introduced or trends change, machine learning incorporates that information to determine the new norm without people needing to go back in and reprogram baselines or key performance indicators.

This ability to learn and madapt makes it the optimal choice for improvements in ongoing processes, marketing campaigns and continuous customer service improvements.

The machine learning algorithm is mainly divided into:

- Training phase
- Testing phase

Training Phase

Take a randomly selected specimen of mangoes from the market (training data), make a table of all the physical characteristics of each mango, like color, size, shape, grown in which part of the country, sold by which vendor, etc (features), along with the sweetness, juiciness, ripeness of that mango (output variables). Now feed this data to the machine learning algorithm (classification/regression), and it learns a model of the correlation between an average mango's physical characteristics, and it's quality.

Testing Phase

Next time when go shopping, you will measure the characteristics of the mangoes which you are purchasing (test data)and feed it to the Machine Learning algorithm. It will use the model which was computed earlier to predict if the mangoes are sweet, ripe and/or juicy. The algorithm may internally use the rules, similar to the one you manually wrote earlier (e.g., a decision tree). Finally, you can now shop for mangoes with great confidence, without worrying about the details of how to choose the best mangoes.

11.2 Objective

After going through the unit you will be able to:

- Understand briefly about machine learning
- Understand difference between supervised and unsupervised learning
- Learn the concept of Machine Learning and Data Mining, its similarity and dissimilarities

11.3 Supervised Learning

In supervised learning, both the input and desired output are provided and the machine must learn how to map the former to the latter. To accomplish this, the machine is trained on a statistically representative set of example inputs and corresponding outputs.

An example of this could be teaching a machine to recognize a picture of a dog. You would train the machine by showing it pictures of various breeds of dogs, labeled as dogs as compared to pictures of cats, labeled as cats. When it comes across a picture of a dog, it would recognize it as a dog based on the data on which it had been trained. It does this by computing the specific characteristics, or features, of the input image and comparing them to the features of labeled images or objects.

One pro for this approach is that the system can be better controlled and the accuracy typically increases with the number of labeled examples or patterns provided. On the flip side, qualified people need to label the examples or patterns to be used for training. This can be very time consuming and labor-intensive and there are limits to scalability with this approach. The majority of practical machine learning uses supervised learning. Supervised learning is where you have input variables (X) and an output variable (Y) and you use an algorithm to learn the mapping function from the input to the output.

$$Y = f(X) \qquad (11.1)$$

The goal is to approximate the mapping function so well that when you have new input data(X) that you can predict the output variables (Y) for that data. It is called supervised learning, because

the process of algorithm learning from the training dataset can be thought of as a teacher supervising the learning process. We know the correct answers; the algorithm iteratively makes predictions on the training data and is corrected by the teacher. Learning stops when the algorithm achieves an acceptable level of performance. Supervised learning problems can be further grouped into regression and classification problems.

- Classification: A classification problem is when the output variable is a category, such as red or blue or disease and no disease.

- Regression: A regression problem is when the output variable is a real value, such as dollars or weight.

Some common types of problems built on top of classification and regression include recommendation and time series prediction respectively.
Some popular examples of supervised machine learning algorithms are:

- Linear regression for regression problems.
- Random forest for classification and regression problems.
- Support vector machines for classification problems.

11.4 Unsupervised Learning

In unsupervised learning, the machine is not provided labeled examples or previous patterns on which to base the analysis of the data inputs. The machine must uncover patterns and draw inferences by itself, without having the correct answers. It will classify or cluster data by discovering the similarity of features on its own. Using unsupervised learning, the machine would be fed millions of pictures of dogs, without labeling them as dogs. It would use the text in the web copy or captions associated with the pictures to decipher clues, particularly noting that the word dog often showed up in the various texts, and would label the photos as dogs.

A pro here is that you do not need a person to label the examples or patterns and therefore people are not involved in the training. This can also be a con because there is no human interaction to train the machine and initially it will not know if the classifications it makes are right or wrong. There can be more erroneous results initially. The patterns and clusters discovered may or may not be of value to you – this again can be a pro or con. You may discover trends you were not looking for, but you may also not get the results you desire.

Unsupervised learning is where you only have input data (X) and no

corresponding output variables. The goal of unsupervised learning model the underlying structure or distribution in the data to learn about the data. is to more

These are called unsupervised learning because unlike supervised learning above there is no correct answer and there is no teacher. Algorithms are left to their own devises to discover and present the interesting structure in the data. Unsupervised learning problems can be further grouped into clustering and association problems.

- Clustering: A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.
- Association: An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy A also tend to buy B.

Some popular examples of unsupervised learning algorithms are:

- *k*-means for clustering problems.
- A priori algorithm for association rule learning problems.

11.5 Machine Learning and Data Mining

Machine learning involves the study of algorithms that can extract information automatically. The source for machine learning is also data (technically says databases), basically it involves two sets of data training data as well as test data. Usually, machine learning uses data mining techniques and another learning algorithm to build models of what is happening behind some data so that it can predict future outcomes.

Data mining refers to extracting knowledge from a large amount of data, in the other way we can say data mining is the process to discover various types of patterns that are inherited in the data and which are accurate, new and useful. It is an iterative process of creating a predictive and descriptive model, by uncovering previously unknown trends and patterns in vast amounts of data to support decision making. Data mining is the subset of business analytics, it is similar to experimental research. The origins of data mining are databases, statistics. Whereas machine learning involves the algorithm that improves automatically through experience based

on data. In simple word, we can say that machine learning is a way to discover a new algorithm from the experience

1. To implement data mining techniques, it used two-component first one is the database and the second one is machine learning. The Database offers data management techniques while machine learning offers data analysis techniques. But to implement machine learning techniques it used algorithms.
2. Data mining uses more data to extract useful information and that particular data will help to predict some future outcomes for example in a sales company it uses last year data to predict this sale but machine learning will not rely much on data it uses algorithms, for example, OLA, UBER machine learning techniques to calculate the ETA for rides.
3. Self-learning capacity is not present in data mining; it follows the rules and predefined. It will provide the solution for a particular problem but machine learning algorithms are self-defined and can change their rules as per the scenario, it will find out the solution for a particular problem and it resolves it in its way.
4. The main and foremost difference between data mining and machine learning is, without the involvement of human data mining can't work but in machine learning human effort is involved only the time when algorithm is defined after that it will conclude everything by own means once implemented forever to use but this is not the case with data mining.
5. The result produces by machine learning will be more accurate as compared to data mining since machine learning is an automated process.
6. Data mining uses the database or data warehouse server, data mining engine and pattern evaluation techniques to extract useful information whereas machine learning uses neural networks, predictive model and automated algorithms to make the decisions.

11.6 Check your progress questions

1. Define Machine Learning.
2. Define Supervised Machine Learning.
3. Define Unsupervised Machine Learning.

11.7 Answer to check your progress questions

1. Machine Learning is a concept that allows the machine to learn from examples and experience, and that too without being explicitly programmed. So instead of writing the code, we feed data to the generic algorithm, and the algorithm/machine builds the logic based on the given data.
2. In supervised learning, both the input and desired output are provided and the machine must learn how to map the former to the latter. To

accomplish this, the machine is trained on a statistically representative set of example inputs and corresponding outputs.

3. In unsupervised learning, the machine is not provided

labeled examples or previous patterns on which to base the analysis of the data inputs.

4.The machine must uncover patterns and draw inferences by itself, without having the correct answers. It will classify or cluster data by discovering the similarity of features on its own.

11.8 Summary

The main and foremost difference between data mining and machine learning is, without the involvement of human data mining can't work but in machine learning human effort is involved only the time when algorithm is defined after that it will conclude everything by own means once implemented forever to use but this is not the case with data mining. Supervised learning is the process of building classification models using data instances of known origin. Unsupervised learning is the class label of each training tuple is not known.

11.9 Keywords

- **Machine Learning** - Machine learning uses self-learning algorithms to improve its performance at a task with experience over time.
- **Training Phase** - Feed the training data set to the machine learning algorithm (classification/regression), and it learns a model of the correlation between the attributes.
- **Testing Phase** - Test dataset is used to provide an unbiased evaluation of a final model fit on the training dataset.

11.10 Self Assessment Questions and Exercises

1. Explain about machine learning and data mining.
2. Briefly discuss on supervised learning.
3. Write notes on unsupervised learning.

11.11 Further Reading

1. Arun K Pujari, Data Mining Techniques, Universities Press
2. Poonkuzhali. S, Saravanakumar. C, Data Warehousing & Data Mining, Charulatha Publications.
3. Charu C. Aggarwal, Chandan K. Reddy, Data Clustering: Algorithms and Applications, CRC Press
4. Pang-Ning Tan, Vipin Kumar, Michael Steinbach, Introduction to Data Mining, Pearson.
5. Rao. N. Raghavendra, Global Virtual Enterprises in Cloud Computing Environments, United States of America by IGI Global.
6. https://en.wikipedia.org/wiki/Training,_validation,_and_test_sets

UNIT – 12

NEURAL NETWORKS

Structure

12.1 Introduction

As a child, we used to learn things with the help of our elders, which includes our parents or teachers. Then later by self-learning or practice, we keep learning throughout our life. Scientists and researchers are also making the machine intelligent, just like a human being, and ANN plays a very important role in the same due to the following reasons:

- With the help of neural networks, we can find the solution
- of such problems for which algorithmic method is expensive
- or does not exist.
- Neural networks can learn by example, hence we do not
- need to program it at much extent.
- Neural networks have accuracy and significantly fast speed
- than conventional speed.

Neural networks are parallel computing devices, which are an attempt to make a computer model of the brain. The main objective is to develop a system to perform various computational tasks faster than traditional systems.

12.2 Objective

After going through the unit you will be able to understand the concept of neural network and genetic algorithms.

12.3 Uses of Neural Network

The followings are some of the areas, where ANN is being used. It suggests that ANN has an interdisciplinary approach in its development and applications.

Speech Recognition

Speech occupies a prominent role in human-human interaction. Therefore, it is natural for people to expect speech interfaces with computers. In the present era, for communication with machines, humans still need sophisticated languages that are difficult to learn and use. To ease this communication barrier, a simple solution could be communication in a spoken language that is possible for the machine to understand.

Great progress has been made in this field; however, still, such kinds of systems are facing the problem of limited vocabulary or grammar along with the issue of retraining of the system for different speakers in different conditions.

ANN is playing a major role in this area. Following ANNs have been used for speech recognition:

- Multilayer networks
- Multilayer networks with recurrent connections
- Kohonen self-organizing feature map

The most useful network for this is Kohonen Self-Organizing feature map, which has its input as short segments of the speech waveform. It will map the same kind of phonemes as the output array, called feature extraction technique. After extracting the features, with the help of some acoustic models as back-end processing, it will recognize the utterance.

Character Recognition

It is an interesting problem that falls under the general area of Pattern Recognition. Many neural networks have been developed for automatic recognition of handwritten characters, either letters or digits. Following are some ANNs which have been used for character recognition:

- Multilayer neural networks such as Back-propagation neural networks.

Though back-propagation neural networks have several hidden layers, the pattern of connection from one layer to the next is localized. Similarly, neocognitron also has several hidden layers and its training is done layer by layer for such kind of applications.

Signature Verification Application

Signatures are one of the most useful ways to authorize and authenticate a person in legal transactions. The Signature verification technique is a non-vision based technique.

For this application, the first approach is to extract the feature or rather the geometrical feature set representing the signature. With these feature sets, we have to train the neural networks using an efficient neural network algorithm. This trained neural network will classify the signature as being genuine or forged under the verification stage.

Human Face Recognition

It is one of the biometric methods to identify the given face. It is a typical task because of the characterization of "non-face" images. However, if a neural network is well trained, then it can be divided into two classes namely images having faces and images that do not have faces.

First, all the input images must be preprocessed. Then, the dimensionality of that image must be reduced. And, at last, it must be classified using neural network training algorithm. Following neural networks are used for training purposes with preprocessed image:

- Fully-connected multilayer feed-forward neural network trained with the help of the back-propagation algorithm.
- For dimensionality reduction, Principal Component Analysis (PCA) is used.

12.4 Working and Neural Network

Components in Neural Network

A typical neural network has anything from a few dozen to hundreds, thousands, or even millions of artificial neurons called units arranged in a series of layers, each of which connects to the layers on either side. Some of them, known as input units, are designed to receive various forms of information from the outside world that the network will attempt to learn about, recognize, or otherwise process.

Other units sit on the opposite side of the network and signal how it responds to the information it's learned; those are known as output units. In between the input units and output units are one or more layers of hidden units, which, together, form the majority of the artificial brain.

Most neural networks are fully connected, which means each hidden unit and each output unit is connected to every unit in the layers either side. The connections between one unit and another are represented by a number called weight, which can be either positive (if one unit excites another) or negative (if one unit suppresses or inhibits another). The higher the weight, the more influence one unit has on another. (This corresponds to the way actual brain cells trigger one another across tiny gaps called synapses.)

Learning Things in Neural Network

Information flows through a neural network in two ways. When it's learning (being trained) or operating normally (after being trained), patterns of information are fed into the network via the input units, which trigger the layers of hidden units, and these, in turn, arrive at the output units. This common design is called a feed-forward network.

Not all units "fire" all the time. Each unit receives inputs from the units to its left, and the inputs are multiplied by the weights of the connections they travel along. Every unit adds up all the inputs it receives in this way and (in the simplest type of network) if the sum is more than a certain threshold value, the unit "fires" and triggers the units it's connected to (those on its right).

For a neural network to learn, there has to be an element of feedback involved—just as children learn by being told what they're doing right or wrong. In fact, we all use the feedback, all the time.

Neural networks learn things in the same way, typically by a feedback process called Backpropagation. This involves comparing the output a network produces with the output it was meant to produce and

using the difference between them to modify the weights of the connections between the units in the network, working from the output units through the hidden units to the input units, going backward, in other words.

In time, Backpropagation causes the network to learn, reducing the difference between actual and intended output to the point where the two exactly coincide, so the network figures things out exactly as it should.

Working Principle

Once the network has been trained with enough learning examples, it reaches a point where you can present it with an entirely new set of inputs it's never seen before and see how it responds.

For example, suppose you've been teaching a network by showing it lots of pictures of chairs and tables, represented in some appropriate way it can understand, and telling it whether each one is a chair or a table. After showing it, let's say, 25 different chairs and 25 different tables, you feed it a picture of some new design it's not encountered before, let's say a chaise longue, and see what happens. Depending on how you've trained it, it'll attempt to categorize the new example as either a chair or a table, generalizing its experience, just like a human.

Now, you've taught a computer how to recognize furniture! That doesn't mean to say a neural network can just "look" at pieces of furniture and instantly respond to them in meaningful ways; it's not behaving like a person.

Consider the example we've just given: the network is not looking at pieces of furniture. The inputs to a network are essentially binary numbers: each input unit is either switched on or switched off. So if you had five input units, you could feed in information about five different characteristics of different chairs using binary (yes/no) answers. The questions might be 1) Does it have a back? 2) Does it have a top? 3) Does it have soft upholstery? 4) Can you sit on it comfortably for long periods? 5) Can you put lots of things on top of it?

A typical chair would then present as Yes, No, Yes, Yes, No or 10110 in binary, while a typical table might be No, Yes, No, No, Yes or 01001. So, during the learning phase, the network is simply looking at lots of numbers like 10110 and 01001 and learning that some mean chair (which might be an output of 1) while others mean table (output of 0).

12.5 Genetic Algorithm

Nature has always been a great source of inspiration to all mankind. Genetic Algorithms (GAs) are search-based algorithms based on the concepts of natural selection and genetics. GAs is a subset of a much larger branch of computation known as Evolutionary Computation.

GAs was developed by John Holland and his students and colleagues at the University of Michigan, most notably David E. Goldberg and has since been tried on various optimization problems with a high degree of success.

Genetic algorithms attempt to incorporate ideas of natural evolution. In general, genetic learning starts as follows. An initial population is created consisting of randomly generated rules. Each rule can be represented by a string of bits.

As a simple example, suppose that samples in a given training set are described by two Boolean attributes, A_1 and A_2, and that there are two classes, C_1 and C_2. The rule "*IF* A_1 *AND NOT* A_2 *THEN* C_2" can be encoded as the bit string "100," where the two leftmost bits represent attributes A_1 and A_2, respectively, and the rightmost bit represents the class. Similarly, the rule "*IF NOT* A_1 *AND NOT* A_2 *THEN* C_1" can be encoded as "001."

If an attribute has k values, where $k > 2$, then k bits may be used to encode the attribute's values. Classes can be encoded in a similar fashion. Based on the notion of survival of the fittest, a new population is formed to consist of the fittest rules in the current population, as well as the offspring of these rules. Typically, the fitness of a rule is assessed by its classification accuracy on a set of training samples.

Offspring are created by applying genetic operators such as crossover and mutation. In the crossover, substrings from pairs of rules are swapped to form new pairs of rules. In mutation, randomly selected bits in a rule's string are inverted. The process of generating new populations based on prior populations of rules continues until a population, P, evolves where each rule in P satisfies a prespecified fitness threshold.

Genetic algorithms are easily parallelizable and have been used for classification as well as other optimization problems. In data mining, they may be used to evaluate the fitness of other algorithms.

A genetic algorithm is a computational model consisting of five parts:

- A staring set of individuals, P.
- **Crossover** technique to combine two parents to create offspring

- **Mutation**: randomly change an individual
- **Fitness**: determine the best individuals
- **The Algorithm** which approves the crossover and mutation techniques to P iteratively using the fitness function to determine the best individuals in P to keep.

Advantage

- Easily parallelized.

Disadvantages

- Difficult to understand and explain to end-users.
- The Abstraction of the problem and methods to represent individuals is quite difficult.
- Determining fitness function is difficult.
- Determining how to perform crossover and mutation is difficult.

12.6 Check your progress questions

1. Define ANN.
2. Write down the uses of neural network.
3. Define Genetic Algorithm.

12.7 Answer to check your progress questions

1. ANN stands for Artificial Neural Network is non-linear predictive models that learn through training and resemble biological neural networks in structure.
2. Neural network are used in various application such as Speech Recognition, Character Recognition, Signature verification application, and Human face recognition
3. Genetic Algorithm is the Optimization techniques that use processes such as genetic combination, mutation, and natural selection in a design based on the concepts of natural evolution.

12.8 Summary

Neural Network is a set of interconnected nodes designed to imitate the functioning of the human brain. A neural network architecture where all weights at one layer are directed toward nodes at the next network layer, the weights do not cycle back as inputs to previous layer is known as feed-forward neural network.

12.9 Keywords

- **Backpropagation Learning** – It is a training method used with many feed-forward networks that works by making modifications in weight values starting at the output layer moving backward through the hidden layer.
- **Crossover** – A genetic algorithm operation that creates new population elements by combining parts of two or more elements from the current population.
- **Mutation** – A genetic learning operation that creates a new population element by randomly modifying a portion of an existing element.

12.10 Self Assessment Questions and Exercises

1. Write short notes on uses of neural network.
2. Explain about the working principle of neural network.
3. Describe about genetic algorithm.

12.11 Further Reading

1. Poonkuzhali. S, Saravanakumar. C, Data Warehousing & Data Mining, Charulatha Publications.
2. Jiawei Han, Micheline Kambar, Jian Pei, Data mining concepts and techniques, Morgan Kaufmann is an imprint of Elsevier.
3. Arun K Pujari, Data Mining Techniques, Universities Press
4. Bharat Bhushan Agarwal, Sumit Prakash Tayal, Data Mining and Data Warehousing, University Science Press.
5. Pang-Ning Tan, Vipin Kumar, Michael Steinbach, Introduction to Data Mining, Pearson.

BLOCK 5
WEB MINING

UNIT –13
INTRODUCTION

Structure

13.1 Introduction

Web mining is an application of data mining techniques to find information patterns from the web data. Web mining helps to improve the power of web search engine by identifying the web pages and classifying the web documents. Web mining is very useful to e-commerce websites and e-services. Web mining is a branch of data mining concentrating on the World Wide Web as the primary data source, including all of its components from Web content, server logs to everything in between. The contents of data

mined from the Web may be a collection of facts that Web pages are meant to contain, and these may consist of text, structured data such as lists and tables, and even images, video and audio.

13.2 Objectives

After going through the unit you will be able to;

- Know the fundamentals Web Mining
- Understand the basic concepts of all types of Web mining

13.3 Web Content Mining

This is the process of mining useful information from the contents of Web pages and Web documents, which are mostly text, images and audio/video files. Techniques used in this discipline have been heavily drawn from natural language processing (NLP) and information retrieval. Web content mining can be used for mining of useful data, information and knowledge from web page content.

Web structure mining helps to find useful knowledge or information pattern from the structure of hyperlinks. Due to heterogeneity and absence of structure in web data, automated discovery of new knowledge pattern can be challenging to some extent. Web content mining performs scanning and mining of the text, images and groups of web pages according to the content of the input (query), by displaying the list in search engines.

For example: If an user wants to search for a particular book, then search engine provides the list of suggestions.

13.4 Web Structure Mining

This is the process of analyzing the nodes and connection structure of a website through the use of graph theory. There are two things that can be obtained from this: the structure of a website in terms of how it is connected to other sites and the document structure of the website itself, as to how each page is connected. The web structure mining can be used to discover the link structure of hyperlink. It is used to identify that the web pages are either linked by information or direct link connection. The purpose of structure mining is to produce the structural summary of website

and similar web pages.
Example: Web structure mining can be very useful to companies to determine the connection between two commercial websites.

13.5 Web Usage Mining

This is the process of extracting patterns and information from server logs to gain insight on user activity including where the users are from, how many clicked what item on the site and the types of activities being done on the site. Web usage mining is used for mining the web log records (access information of webpages) and helps to discover the user access patterns of web pages. Web server registers a web log entry for every web page. Analysis of similarities in web log records can be useful to identifythe potential customers for e-commerce companies.

13.6 Text mining

Text mining, also known as text analysis, is the process of transforming unstructured text into meaningful and actionable information. By identifying topics, patterns, and relevant keywords, text mining allows you to obtain valuable insights without needing to go through all your data manually. Thanks to text mining, businesses are being able to analyze complex and large sets of data in a simple, fast and effective way. At the same time, companies are taking advantage of this powerful tool to reduce some of their manual and repetitive tasks, saving their teams precious time and allowing customer support agents to focus on what they do best.
The five fundamental steps involved in text mining are:

1. Gathering unstructured data from multiple data sources like plain text, web pages, pdf files, emails, and blogs, to name a few.

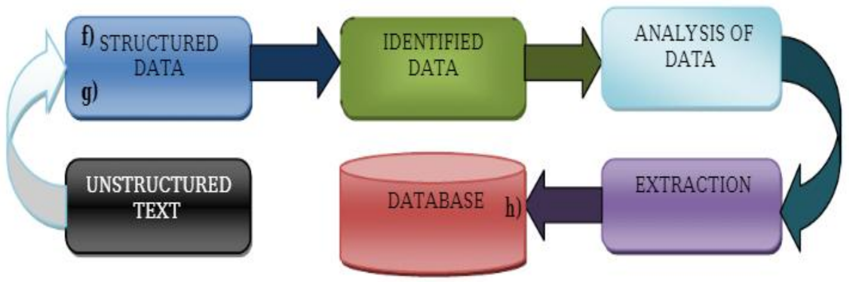
2. Detect and remove anomalies from data by conducting pre processing and cleansing operations. Data cleansing allows you to extract and retain the valuable information hidden within the data and to help identify the roots of specific words.

3. For this, you get a number of text mining tools and text mining applications.

4. Convert all the relevant information extracted from unstructured data into structured formats.

5. Analyze the patterns within the data via the Management Information System (MIS).

- Store all the valuable information into a secure database to drive trend analysis and enhance the decision-making process of the organization.



Text Mining Techniques

Text mining techniques can be understood at the processes that go into mining the text and discovering insights from it. These text mining techniques generally employ different text mining tools and applications for their execution. Now, let us now look at the various text mining techniques:

13.7 Text Clustering

This is Meaning Cloud's solution for **automatic document clustering**, i.e., the task of grouping a set of texts in such a way that texts in the same group (called a cluster) are more similar to each other than to those in other clusters.

The algorithm receives a set of texts and returns the list of detected clusters. Each cluster is assigned a descriptive name, a relevance value (indicating the relative importance of the cluster with respect to all clusters), its size, and the list of elements that are included in the cluster. Each document may be assigned to one or several clusters.

Text clustering may be used for different tasks, such as **grouping**

similar documents (news, tweets, etc.) and the **analysis of customer/employee feedback**, **discovering meaningful implicit subjects** across all documents.**How it works**

Typically, descriptors (sets of words that describe topic matter) are extracted from the document first. Then they are analyzed for the frequency in which they are found in the document compared to other terms. After which, clusters of descriptors can be identified and then auto-tagged. From there, the information can be used in any number of ways. Google's search engine is probably the best and most widely known example. When you search for a term on Google, it pulls up pages that apply to that term, but have you ever wondered how Google can analyze billions of web pages to deliver an accurate and fast result? It's because of text clustering! Google's algorithm breaks down unstructured data from web pages and turns it into a matrix model, tagging pages with keywords that are then used in search results!

Example

To help you understand the process, it's best to visualize an example:

Let's simulate how text clustering would analyze (and tag) this sentence.

First, all punctuation is removed:

let us simulate how text clustering would analyze and tag this sentence

Then, all but the sentence's descriptors are removed:

simulate how text clustering analyze tag sentence

At this point, its harder to visualize as a computer will be assigning each word a weighted value for use in tagging.

13.8 Temporal

Temporal data mining refers to the extraction of implicit, non-trivial, and potentially useful abstract information from large collections of temporal data. Temporal data are sequences of a primary data type, most commonly numerical or categorical values and sometimes multivariate or composite information. Examples of temporal data are regular time series (e.g., stock ticks, EEG), event sequences (e.g., sensor readings, packet traces, medical records, weblog data), and temporal databases (e.g., relations with time stamped tuples, databases with versioning). The common factor of all these sequence types is the total ordering of their elements. They differ on the type of primary information, the regularity of the elements in the sequence, and on whether there is explicit temporal information associated to each element (e.g., timestamps).

13.9 Spatial

Spatial data mining is the application of data mining to spatial models. In spatial data mining, analysts use geographical or spatial information to produce business intelligence or other results. This requires specific techniques and resources to get the geographical data into relevant and useful formats. Challenges involved in spatial data mining include identifying patterns or finding objects that are relevant to the questions that drive the research project. Analysts may be looking in a large database field or other extremely large data set in order to find just the relevant data, using GIS/GPS tools are similar system.

One interesting thing about the term "spatial data mining" is that it is generally used to talk about finding useful and non-trivial patterns in data. In other words, just setting up a visual map of geographic data may not be considered spatial data mining by experts. The core goal of a spatial data mining project is to distinguish the information in order to build real, actionable patterns to present, excluding things like statistical coincidence, randomized spatial modeling or irrelevant results. One way analysts may do this is by combing through data looking for "same- object" or "object-equivalent" models to provide accurate comparisons of different geographic locations

13.10 Visual Data Mining

Visual Data Mining uses data and/or knowledge visualization techniques to discover implicit knowledge from large data sets.

Visual data mining can be viewed as an integration of the following disciplines −

- Data Visualization
- Data Mining

Visual data mining is closely related to the following −

- Computer Graphics
- Multimedia Systems
- Human Computer Interaction
- Pattern Recognition
- High-performance Computing

Generally data visualization and data mining can be integrated in the following ways

Data Visualization

The data in a database or a data warehouse can be viewed in several visual forms that are listed below −

- Boxplots
- 3-D Cubes
- Data distribution charts
- Curves
- Surfaces
- Link graphs etc.

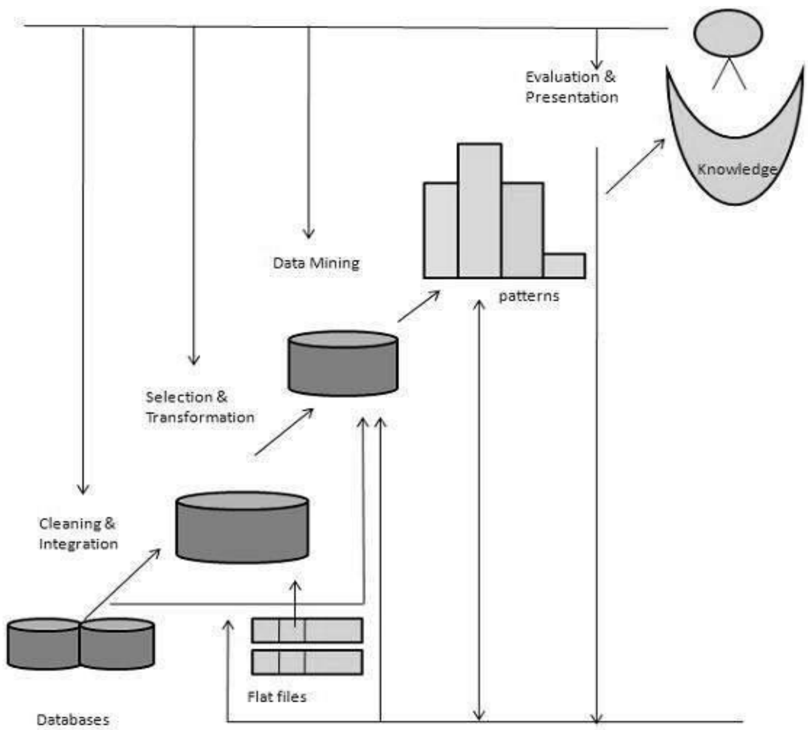
Data Mining Result Visualization − Data Mining Result Visualization is the presentation of the results of data mining in visual forms. These visual forms could be scattered plots, boxplots, etc.

Data Mining Process Visualization − Data Mining Process Visualization presents the several processes of data mining.
It allows the users to see how the data is extracted. It also allows the users to see from which database or data warehouse the data is cleaned, integrated, preprocessed, and mined.

13.11 Knowledge Mining

Some people don't differentiate data mining from knowledge discovery while others view data mining as an essential step in the process of knowledge discovery. Here is the list of steps involved in the knowledge discovery process −

- **Data Cleaning** − In this step, the noise and inconsistent data is removed.
- **Data Integration** − In this step, multiple data sources are combined.
- **Data Selection** − In this step, data relevant to the analysis task are retrieved from the database.
- **Data Transformation** − In this step, data is transformed or consolidated into forms appropriate for mining by performing summary or aggregation operations.
- **Data Mining** − In this step, intelligent methods are applied in order to extract data patterns.
- **Pattern Evaluation** − In this step, data patterns are evaluated.
- **Knowledge Presentation** − In this step, knowledge is represented.



The above diagram shows the process of knowledge discovery

13.12 Check Your Progress Questions

1) What are areas of Text mining
2) What is web content mining?
3) What are the steps involved in Knowledge mining?

13.13 Answers to Check Your Progress Questions

1. Information Extraction, Natural Language Processing, Data Mining, Information Retrieval
2. Web content mining can be used for mining of useful data, information and knowledge from web page content. Web structure mining helps to find useful knowledge or information pattern from the structure of hyperlinks. Due to heterogeneity and absence of structure in web data, automated discovery of new knowledge pattern can be challenging to some extent.
3. Data Cleaning, Data Integration, Data Selection, Data Transformation, DataMining, Pattern Evaluation, Knowledge Presentation

13.14 Summary

Web Mining is moving the World Wide Web towards a more useful environment in which users can quickly and easily find the information they need. Large amount of text documents, multimedia files and images are available in the web and it is still increasing. Data mining is the form of extracting data's available in the internet. Web mining is a part of data mining. Web mining is used to discover and extract information from Web-related data

sources such as Web documents, Web content, hyperlinks and server logs. The term Web mining has been used in three distinct ways. The first, called Web content mining is the process of information discovery from sources across the World Wide Web.

The second, called Web structure mining is the process of analyzing the relationship between Web pages linked by information or direct link connection through the use of graph theory. The third, called Web usage mining is the process of extracting patterns and information from server logs to gain insight on user activity.

13.15 Key Words

MIS - Management Information Systems
Cluster- the task of grouping a set of texts in such a way that texts in the same group

13.16 Self Assessment Questions and Exercises

1. Explain in detail about the Web usage Mining
2. Short notes on Visual Data Mining?
3. What are the benefits Temporal Mining?

13.17 Further Readings

1. R. Krishnamoorthy and S. Prabhu, Internet and Java Programming, New Age International Publishers, 2004
2. Programming with Java, 4e, E. Balagurusamy, Tata McGraw-Hill, 2010.
3. Deitel, Deitel and Nieto, Internet and World Wide Web – How to program, Pearson Education, 2000.
4. Naughton and H.Schildt, Java 2 - The complete reference, Tata McGraw-Hill, Fourth edition, 2006.
5. Elliotte Rusty Harold, Java Network Programming, O’Reilly Publishers, 2000.

6. B.Mohamal Ibrahim , Java : J2SE – A Practical Approach, Firewall media, 2006.
7. Cay S. Horstmann, Gary Cornell, Core Java, Volume I and II, 5th Edition, Pearson Education, 2003.
8. Topley, J2ME in A Nutshell, O'Reilly Publishers, 2002.
9. Hunt, Guide to J2EE Enterprise Java, Springer Publications, 2004.
10. Ed Roman, Enterprise Java Beans, Wiley Publishers, 1998.
11. Agrawal R., Imielinski T., and Swami A.N. Mining association rules between sets of items in large databases.
12. In Proc. ACM SIGMOD Int. Conf. on Management of Data, 1993, pp.207–216.Google Scholar
13. Agrawal R. and Srikant R. Fast algorithms for mining association rules in large databases. In Proc. 20th Int. Conf. on Very Large Data Bases, 1994, pp. 487–499.
    Google Scholar
14. Agrawal R. and Srikant R. Mining sequential patterns. In Proc. 11th Int. Conf. on Data Engineering, 1995, pp. 3–14.Google Scholar

UNIT 14
TOOLS AND TECHNIQUES

Structure

14.1 Introduction

This is a customization tools, which is free to use. It includes visualization and predictive analysis and modelling techniques, clustering, association, regression and classification.

14.2 Objectives

After going through the unit you will be able to;

- Understand Weka Tool
- Know about Rapidminer
- Learn about Matlab

14.3 Using Weka

Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes

Weka website (Latest version 3.6):-http://www.cs.waikato.ac.nz/ml/weka/

Weka Manual: − http://transact.dl.sourceforge.net/sourceforge/weka/WekaManual-3.6.0.pdf

Datasets in Weka

Each entry in a dataset is an instance of the java class: weka.core.Instance Each instance consists of a number of attributes.

Attributes

Nominal: one of a predefined list of values

e.g. red, green, blue

Numeric: A real or integer number

String: Enclosed in “double quotes”

Date

Relational

Classifiers in Weka

Learning algorithms in Weka are derived fromλ the abstract class:

- weka.classifiers.Classifier
- Simple classifier: ZeroR
- Just determines the most common class
- Or the median (in the case of numeric values) –
- Tests how well the class can be predicted without considering other attributes

Can be used as a Lower Bound on Performance

14.4 Rapidminer and matlab

RapidMiner Studio is a **powerful data mining tool** for rapidly building predictive models. This all-in-one tool features hundreds of data preparation and machine learning algorithms to support all your data mining projects. Over 30,000+ organizations in every industry to drive revenue, reduce costs, and avoid risks. Get started on your data mining project today by downloading RapidMiner Studio

Application & Interface

- Easy to use visual environment for building analytics processes:
    - Graphical design environment makes it simple and fast to design better models

  - Visual representation with Annotations facilitates collaboration among all stakeholders

- Every analysis is a process, each transformation or analysis step is an operator, making design fast, easy to understand, and fully reusable

- Guided process design leveraging the Wisdom of Crowds, i.e. the knowledge and the best practices of more than 200,000 users in the RapidMiner community
  - Operator recommender suggesting next steps
  - Parameter recommender indicating which parameters to change & to which values

- Convenient set of data exploration tools and intuitive visualizations

- Share your feedback within the product from the tutorials or help panel

- More than 1500 operators for all tasks of data transformation and analysis

- Support for scripting environments like R, or Groovy for ultimate extensibility

- Seamlessly access and use of algorithms from H2O, Weka and other third-party libraries

- Transparent integration with RapidMiner Server to automate processes for data transformation, model building, scoring and integration with other applications

- Extensible through open platform APIs and a Marketplace with additional functionality

- Powerful Global Search sifts through repositories to quickly retrieve anything, including processes, models, operators, extensions and even UI actions

Process:

Data Access & Management

Data Exploration

Data Prep

Modeling

Validation

Scoring

Automation & Process Control

MATLAB Data Analysis

- Preparing Data
- Basic Fitting
- Correlation

Statistics Toolbox

- Probability Distributions
- Descriptive Statistics
- Linear & Nonlinear Models
- Hypothesis Tests
- Statistical Plots

14.5 Check Your Progress Questions

1. What do you meant Weka Tool

2. What are the Process involved in Rapidminer

14.6 Answers to Check Your Progress Questions

1. Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes.

2. Data Access & Management
    - Data Exploration
    - Data Prep
    - Modeling
    - Validation
    - Scoring
    - Automation & Process Control

14.7 Summary

This is very popular since it is a readymade, open source, no-coding required software, which gives advanced analytics. Written in Java, it incorporates multifaceted data mining functions such as data pre-processing, visualization, predictive analysis, and can be easily integrated with WEKA and R-tool to directly give models from scripts written in the former two. Besides the standard data mining features like data cleansing, filtering, clustering, etc, the software also features built-in templates, repeatable work flows, a professional visualisation environment, and seamless integration with languages like Python and R into work flows that aid in rapid prototyping.

14.8 Key Words

-Weka is a collection of machine learning algorithms for data mining tasks
-RapidMiner is a data science software platform developed by the company of the same name that provides an integrated environment for data preparation, machine learning, deep learning, text mining, and predictive analytics.

14.9 Self-Assessment Questions and Exercises

1. How to Work with Rapidminer?
2. Explain operations in Weka Tool.

14.10 Further Readings

1. https://securityonline.info/8-best-open-source-data-mining-tools-weka-rapid-miner-orange-knime-jhepwork-apache-mahout-elki rattle/
2. https://www.cs.waikato.ac.nz/ml/weka/
3. http://matlabdatamining.blogspot.com/

***

Time : 3 hours   Max Marks :75

PART - A (10 x 2=20 Marks)

Answer all questions.

1. What is Data Warehousing?
2. What are two types of Logical Extraction Method?
3. Define Data Mining
4. Define Data transformation.
5. Define Association Rule.
6. Define Classification
7. Define Clustering.
8. Define Machine Learning.
9. What are the benefits Temporal Mining?
10. What is Rapidminer?

PART - B (5 x 5 Marks = 25 Marks)

Answer all questions choosing either (a) or (b)

.

11. (a) Explain case study about Data warehousing in IT.

Or

(b) Write short note on Data warehousing Operating System

12. (a) Write short notes on different forms of knowledge.

Or

(b)Write short notes on types of data.

13. (a) Explain about partition algorithm with example.

Or

(b) Write short notes on Classification by Back Propagation.

14. (a) Write short notes Machine Learning and data mining.

Or

(b) Briefly discuss the Genetic Algorithm.

15. (a) Write short notes on Web Mining

Or

(b) What are the Process involved in Rapidminer

Part – C (3 x 10 = 30 Marks)

Answer any three questions.

16. Explain Warehouse Schmea?
17. Describe about Data Exploration.
18. Describe about A priori algorithm with an example.
19. Explain in detail about categorical clustering algorithm.
20. Explain Weka Tool Opertions?